2019

Grounding deep models of visual data

BOSTON UNIVERSITY

GRADUATE SCHOOL OF ARTS AND SCIENCES

Dissertation

GROUNDING DEEP MODELS OF VISUAL DATA

by

SARAH ADEL BARGAL

B.Sc., Kuwait University, Kuwait, 2005
M.Sc., The American University in Cairo, Egypt, 2007

Submitted in partial fulfillment of the

requirements for the degree of

Doctor of Philosophy

2019

Approved by

First Reader

Stan Sclaroff, PhD
Professor of Computer Science

Second Reader

Margrit Betke, PhD
Professor of Computer Science

Third Reader

Kate Saenko, PhD
Associate Professor of Computer Science

Acknowledgments

Foremost, I would like to thank my advisor Prof. Stan Sclaroff for his tremendous and continuous support during my PhD studies. I am grateful for all the time and effort he invested in my professional growth and career development, and I cannot thank him enough for his patience and motivation. I really appreciate how he supported my ideas, and how available, welcoming, and friendly he has been during this journey. Being an exceptionally inspiring researcher, teacher, and advisor, I have learned a lot from him. I have also learned many values by experiencing how humble and remarkably respectful he is towards his students and collaborators. I consider myself very lucky to have been his student, and this has greatly influenced the researcher and person I am today.

I would like to thank my committee members Prof. Margrit Betke, Prof. Kate Saenko, Prof. Aude Oliva, and Prof. Evimaria Terzi for their valuable feedback on my work and their valuable career advice. I would also like to thank Dr. Rana el Kaliouby for sparking my interest in Computer Vision and Machine Learning.

I would like to extend my gratitude to all my dear collaborators for all the great work we accomplished together, for their complementary skills, and for their splendid team spirit. I would like to specifically thank Prof. Vittorio Murino, Dr. Andrea Zunino, Dr. Fatih Cakir, Dr. Kun He, and Dr. Shugao Ma for their great collaboration on multiple research projects and Dr. Jianming Zhang for his great mentoring over the years.

I am very grateful for all the opportunities I have been presented with being part of the Image and Video Computing Group, and part of the Computer Science Department at Boston University. I am also very grateful for all my lab mates who have been a great source of support and inspiration. I thank them for their great company, for the numerous research and career planning discussions, and most importantly, for all the fun times we have had together.

I would like to thank my dear family to whom I am forever indebted. I thank my parents Dr. Adel Bargal and Dr. Hemmat Attia for their continuous support and motivation and for who they brought us up to be, and thank my lovely sisters Basma and Salma for being the wonderful supportive friends they are. I would like to thank my baby girl Fayrouz for filling my life with joy and for taking my time management skills to a whole new level :) I would like to immensely thank Ahmad my husband and his lovely family for their encouragement and support. And last, but not least, I owe the perseverance I needed to complete this journey to my dear grandmother, may she rest in peace.

This work was supported in part by NSF grants 1551572 and 1029430, an IBM PhD Fellowship, a Hariri Graduate Fellowship, gifts from Adobe and NVidia, and Intelligence Advanced Research Projects Activity (IARPA) via Department of Interior/ Interior Business Center (DOI/IBC) contract number D17PC00341. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. Disclaimer: The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of IARPA, DOI/IBC, or the U.S. Government.

GROUNDING DEEP MODELS OF VISUAL DATA

SARAH ADEL BARGAL

Boston University, Graduate School of Arts and Sciences, 2019

Major Professor: Stan Sclaroff, Professor of Computer Science

ABSTRACT

Deep models are state-of-the-art for many computer vision tasks including object classification, action recognition, and captioning. As Artificial Intelligence systems that utilize deep models are becoming ubiquitous, it is also becoming crucial to explain why they make certain decisions: *Grounding* model decisions. In this thesis, we study: 1) **Improving Model Classification.** We show that by utilizing web action images along with videos in training for action recognition, significant performance boosts of convolutional models can be achieved. Without explicit grounding, labeled web action images tend to contain discriminative action poses, which highlight discriminative portions of a video's temporal progression. 2) **Spatial Grounding.** We visualize spatial evidence of deep model predictions using a discriminative top-down attention mechanism, called Excitation Backprop. We show how such visualizations are equally informative for correct and incorrect model predictions, and highlight the shift of focus when different training strategies are adopted. 3) **Spatial Grounding for Improving Model Classification at Training Time.** We propose a guided dropout regularizer for deep networks based on the evidence of a network prediction. This approach penalizes neurons that are most relevant for model prediction. By dropping such high-saliency neurons, the network is forced to learn alternative paths in order to maintain loss minimization. We demonstrate better generalization ability, an increased utilization of network neurons, and a higher resilience to network compression.

4) **Spatial Grounding for Improving Model Classification at Test Time.** We propose Guided Zoom, an approach that utilizes spatial grounding to make more informed predictions at test time. Guided Zoom compares the evidence used to make a preliminary decision with the evidence of correctly classified training examples to ensure evidence-prediction consistency, otherwise refines the prediction. We demonstrate accuracy gains for fine-grained classification. 5) **Spatiotemporal Grounding.** We devise a formulation that simultaneously grounds evidence in space and time, in a single pass, using top-down saliency. We visualize the spatiotemporal cues that contribute to a deep recurrent neural network's classification/captioning output. Based on these spatiotemporal cues, we are able to localize segments within a video that correspond with a specific action, or phrase from a caption, without explicitly optimizing/training for these tasks.

Contents

List of Tables

List of Figures

List of Abbreviations

AFEW	.....	Acted Facial Expressions in the Wild
AI	.....	Artificial Intelligence
BU	.....	Boston University
*c*EB	.....	contrastive Excitation Backprop
*c*EB-R	.....	contrastive Excitation Backprop for Recurrent Neural Networks
Caltech	.....	California Institute of Technology
CAM	.....	Class Activation Maps
Cifar	.....	Canadian Institute for Advanced Research
c-MWP	.....	Contrastive Marginal Winning Probability
CNN	.....	Convolutional Neural Network
COCO	.....	Common Objects in Context Dataset
conv	.....	convolutional layer
CPU	.....	Central Processing Unit
D	.....	Dimensional
EB	.....	Excitation Backprop
EB-R	.....	Excitation Backprop for Recurrent Neural Networks
ED	.....	Excitation Dropout
fc	.....	fully-connected layer

FER	.....	Facial Expression Recognition
FV	.....	Fisher Vector encoding
Grad-CAM	.....	Gradient-weighted Class Activation Mapping
GT	.....	Ground-Truth
IDT	.....	Improved Dense Trajectories
LSTM	.....	Long Short Term Memory
mAP	.....	mean Average Precision
MRI	.....	Magnetic Resonance Imaging
rand	.....	random
NLP	.....	Natural Language Processing
ReLU	.....	Rectified Linear Unit
ResNet	.....	Residual Network
RGB	.....	Red Green Blue
RISE	.....	Randomized Input Sampling for Explanations
RNN	.....	Recurrent Neural Network
SFEW	.....	Static Facial Expressions in the Wild
SVM	.....	Support Vector Machine
TDD	.....	Trajectory-pooled Deep-convolutional Descriptor
UCF	.....	University of Central Florida
VisDA	.....	Visual Domain Adaptation
VGG	.....	Visual Geometry Group
WTA	.....	Winner-Take-All

Chapter 1

Introduction

Visual grounding is about explaining the evidence, within a visual input, upon which Artificial Intelligence (AI) systems are making their decisions. As AI systems are becoming integrated into crucial applications, it is also becoming crucial to explain why they make certain decisions. It may be obvious why we would need an explanation for why an AI system makes a mistake, but it is equally important to be able to explain why it makes a correct decision.

Autonomous vehicles are one of AI's current crucial applications. Self-driving cars use cameras as one of their sensory input modalities, and use computer vision algorithms to generate predictions from this modality. One important action such a system learns to predict is *PedestrianCrossing* (Figure 1.1(a)). If the computer vision system classifies every instance of its training data for *PedestrianCrossing* correctly only because it models the periodic motion of the legs, then there will clearly be a problem when the pedestrians have their legs completely occluded as demonstrated in the example of Figure 1.1(b). It is therefore essential to visualize why models make certain predictions, whether such predictions are correct or incorrect.

In this thesis, we explore how AI algorithms, particularly deep neural networks, can benefit from an improved generalization ability, first without grounding, then with grounding.

Figure 1.1: Modeling the periodic motion of the human legs for the *PedestrianCrossing* (left) action may not be sufficient for autonomous vehicles. Surprise variations (right) can happen at test time where the legs are completely occluded. Since it is difficult to make sure that all variations are covered by a training set, visualizing evidence of actions can help alleviate this problem.

1.1 Problems of Visual Grounding

In the following sections we will describe the challenges related to the problems of improving model predictions. First, we describe the challenges of training deep models for the action recognition task and propose one way to improve model prediction without grounding. We then demonstrate limitations of such approaches demonstrated in the difficulty of interpretation of resulting predictions. Next, we present how spatial grounding can help interpret and visually explain a model's prediction. Next, we present challenges of improving model predictions and propose techniques to do so utilizing spatial grounding at training and test times. Finally, we present the spatiotemporal grounding challenge, and propose a formulation to extend grounding to become spatiotemporal.

1.1.1 Improving Model Classification

Recently, attempts have been made to collect millions of videos [55, 1] to train CNN models for action classification in videos. However, curating such large-scale video datasets requires immense human labor, and training CNNs on millions of videos demands huge

computational resources. In contrast, collecting action images from the Web is much easier and training on images requires much less computation. In addition, labeled web images tend to contain discriminative action poses, which highlight discriminative portions of a video's temporal progression. Through extensive experiments, we explore the question of whether we can utilize web action images to train better CNN models for action recognition in videos. We collect 23.8K manually filtered images from the Web that depict the 101 actions in the UCF101 action video dataset. We show that by utilizing web action images along with videos in training, significant performance boosts of CNN models can be achieved. We also investigate the scalability of the process by leveraging crawled web images (unfiltered) for UCF101 and ActivityNet. Using unfiltered images we can achieve performance improvements that are on-par with using filtered images. This means we can further reduce annotation labor and easily scale-up to larger problems. We also shed light on an artifact of finetuning CNN models that reduces the effective parameters of the CNN and show that using web action images can significantly alleviate this problem.

Improving a model's classification accuracy has benefits, but some misclassifications will persist. For example, Figure 1.2 shows a sample frame from a *BabyCrawling* video classified as *PushUps*. It is very interesting to compare frames from this misclassified video to frames from training videos of *Pushups*, where we see analogous body poses. It is not possible from the current setup to confirm if the similar body pose is the reason for the misclassification, it may just be that the training data had samples of pushups being performed on a similar carpet as that in (a). In the next section, we explain the problem of spatial grounding of deep models, which could help provide an explanation for such a misclassification.

(a)  (b) 

Figure 1.2: (a) *BabyCrawling* misclassified as *Pushups* (b) Training video frames for *Pushups*. Note: These frames were not randomly selected; they were selected to illustrate visual similarity.

1.1.2 Spatial Grounding: Background and Applications

Deep convolutional neural network models make predictions based on evidence in visual data. Grounding model decisions in visual data has the benefit of being clearly interpretable by humans. Sample spatial grounding for the task of facial emotion recognition is presented in Figure 1.3. The evidence upon which a model participates in the class conditional probability for a specific class is highlighted in the form of a saliency map.

Various methods have been proposed for grounding the prediction of a convolutional neural network. Some grounding techniques assume knowledge of the network architecture and weights such as [157], and others use randomized masks to infer saliency maps from black-box models whose architectures and weights are unknown such as Petsiuk *et al.* [93]. Some grounding techniques use sliding masks together with monitoring of the output class conditional probabilities to predict salient regions [145, 155]. Other techniques rely on error backpropagation [106, 145, 111]. A recent class of approaches alter the network's architecture before creating such saliency maps [15, 156].

Excitation Backprop (EB) is inspired by a top-down human visual attention model, to pass along top-down signals downwards in the network hierarchy via a probabilistic Winner-Take-All process [118]. EB models the top-down attention of a convolutional neural network classifier for generating task-specific attention maps. EB visualizes the evidence

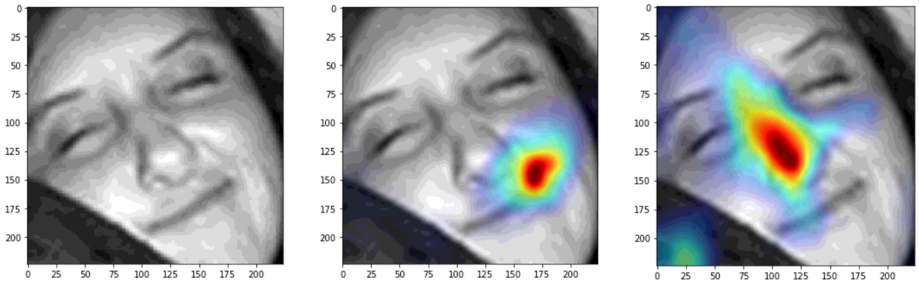


Figure 1.3: In this figure we present how Excitation Backprop can highlight evidence for a ground-truth class (middle; *Happy*), and a non-ground-turth class (right; *Neutral*). Model training: A VGG16 pretrained on VGG-face image dataset [89] was then finetuned on Facial Expression Recognition (FER) Challenge image dataset [10].

of a model's classification decision by computing top-down attention maps using an interpretable probabilistic formulation. It does not require modifying a network's architecture or performing additional training. EB also introduces the concept of contrastive attention, making the top-down attention maps more discriminative for localization purposes.

EB is capable of visualizing evidence for a class whether or not the class is a ground-truth class as demonstrated in Figure 1.3. EB highlights the evidence it would use for making a prediction of the facial emotion *Happy*, and the evidence it would use for making a prediction of the facial emotion *Neutral*.

We demonstrate applications of EB in model interpretation and data annotation assistance for facial expression analysis and medical imaging tasks. We demonstrate how EB can be used to explain model predictions and as a diagnostic tool for model misclassifications. We also demonstrate means of visualizing how deep models differentiate between domains, and how the attention of a model shifts before and after domain adaptation.

1.1.3 Spatial Grounding for Improving Model Classification at Training Time

Deep neural networks optimize for millions of parameters during training time. This huge number of parameters, together with large datasets, and computational resources that allow

training on many epochs of such datasets, can all easily lead to models that overfit the training data and do not generalize well to test examples. Model averaging is a popular regularization technique; however, performing model averaging through training many deep neural networks is a huge waste of resources. A way to execute model averaging that is better-suited for such deep models is: Dropout.

Dropout [40, 112] is a classical regularization technique that is used in many state-of-the-art deep neural networks. Standard Dropout selects a fraction of neurons to randomly drop out by zeroing their forward signal. Curriculum dropout [83], a recent variant of dropout, improves generalization by adjusting the dropout rate during training, answering the question *How many neurons to drop out over time?* Both Standard and Curriculum Dropout select neurons to be dropped randomly. In this work, we target at determining how the dropped neurons are selected, answering the question *Which neurons to drop out?*

We propose a guided dropout regularizer for deep networks that biases the selection of neurons to be dropped-out. Our scheme utilizes the contribution of neurons, *i.e.* evidence, to the prediction made by the network at a certain training iteration. We do so by utilizing the evidence at each neuron to determine the probability of dropout; neuron dropout probability is sampled according to probability defined in the saliency map rather than uniformly at random as in standard dropout. In essence, we dropout with higher probability neurons that contribute more to decision making at training time. This approach penalizes high saliency neurons that are most relevant for model prediction, *i.e.* those having stronger evidence.

By dropping such high-saliency neurons, we deliberately, and temporarily, paralyze/injure neurons such that a deep network is forced to learn alternative paths in order to maintain loss minimization. This results in a plasticity-like behavior, a characteristic of human brains too [38, 109, 80, 79].

We demonstrate better generalization ability, and an increased utilization of network

neurons using several metrics over four image/video recognition benchmarks. We also study network resilience to neuron dropping at test time. We observe that training with Excitation Dropout leads to models that are a lot more robust when layers are compressed by removing units to make models lighter at test time.

1.1.4 Spatial Grounding for Improving Model Classification at Test Time

Spatial grounding is being widely used for many computer vision tasks including spatial semantic segmentation [71, 158, 132], spatial object localization [150, 147], and temporal action localization [8]. However, it has been less exploited for improving model classification. Cao *et al.* [15] use weakly supervised saliency to feedback highly salient regions into the same model that generated them to get more prediction probabilities for the same image and improve classification accuracy at test time. In contrast, we use weakly supervised saliency to question whether the obtained evidence is coherent with the evidence used at training time for correctly classified examples. In Section 1.1.3 we use spatial grounding at training time to improve model classification by dropping neurons corresponding to high-saliency patterns for regularization. In contrast, we now propose an approach to improve model classification at test time.

We propose `Guided Zoom`, an approach that utilizes spatial grounding to make more informed predictions. It does so by making sure the model has "the right reasons" for a prediction, being defined as reasons that are coherent with those used to make similar correct decisions at training time. The reason/evidence upon which a deep neural network makes a prediction is defined to be the spatial grounding, in the pixel space, for a specific class conditional probability in the model output.

`Guided Zoom` estimates how reasonable the evidence used to make a prediction is. In state-of-the-art deep single-label classification models, the top-k ($k = 2, 3, 4, \dots$)

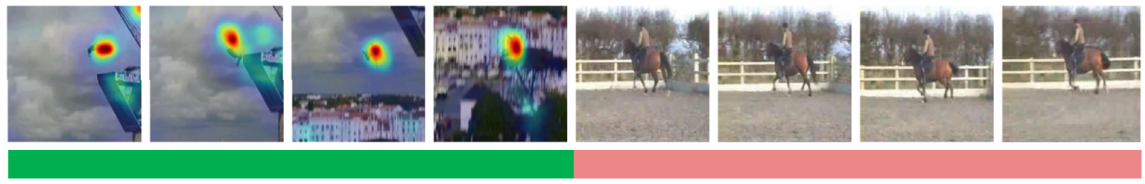
accuracy is usually significantly higher than the top-1 accuracy. This is more evident in fine-grained datasets, where differences between classes are quite subtle. We show that `Guided Zoom` results in the refinement of a model's classification accuracy on three fine-grained classification datasets. We also explore the complementarity of different grounding techniques, by comparing their ensemble to an adversarial erasing approach that iteratively reveals the next most discriminative evidence.

1.1.5 Spatiotemporal Grounding

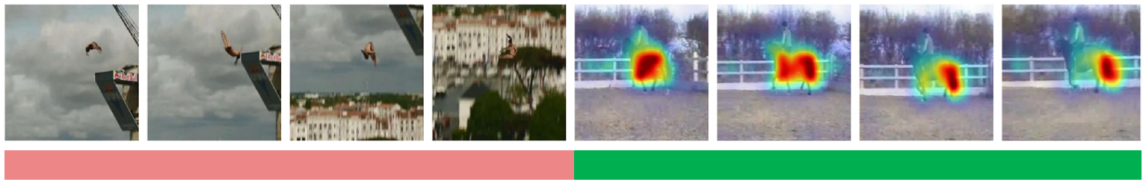
Deep recurrent models are state-of-the-art for many vision tasks including video action recognition and video captioning. Models are trained to caption or classify activity in videos, but little is known about the evidence used to make such decisions. Grounding model decisions in visual data has the benefit of being clearly interpretable by humans. Sample spatiotemporal grounding for the task of action recognition is presented in Figure 1.4.

Grounding decisions made by deep networks has been studied in spatial visual content, giving more insight into model predictions [101, 30, 111, 145, 156, 148]. Such approaches are mainly devised for image understanding and can identify the importance of class-specific image regions by means of saliency maps in a weakly-supervised way. For example, Zhang *et al.* [148] generated class activation maps from any CNN architecture that uses non-linearities producing non-negative activations.

However, such studies are relatively lacking for models of spatiotemporal visual content – videos. Karpathy *et al.* [54] visualized interpretable LSTM cells that keep track of long-range dependencies such as line lengths, quotes, and brackets in a character-based model. Li *et al.* [70] visualized a unit's salience for NLP. Selvaraju *et al.* [103] qualitatively present grounding for captioning and visual question answering in images using an RNN. Ramanishka *et al.* [97] explored visual saliency guided by captions in an encoder-decoder



(a)



(b)

Figure 1.4: Spatiotemporal grounding of (a) the evidence of the action *CliffDiving*, and (b) the evidence of the action *HorseRiding*, in a video containing a concatenation of the two actions. This is a sample result from our formulation devised in Chapter 7.

model. In contrast, our approach models the top-down attention mechanism of CNN-RNN models to produce interpretable and useful task-relevant spatiotemporal saliency maps that can be used for action localization in videos.

We introduce a formulation that simultaneously grounds evidence in space and time, in a single pass, using top-down saliency. We are the first to formulate top-down saliency in deep recurrent models for space-time grounding of videos. We do so using a *single contrastive* Excitation Backprop pass of an already trained model. Although we are not directly optimizing for localization, we show that the internal representation of the model can be utilized to perform coarse localization.

1.2 Contributions

The contributions of this thesis are summarized as follows:

- We show that by utilizing web action images along with videos in training for action recognition, significant performance boosts of convolutional models can be achieved. Without explicit grounding, labeled web action images tend to contain

discriminative action poses, which highlight discriminative portions of a video's temporal progression.

- We visualize grounded spatial evidence of deep model predictions using a discriminative top-down attention mechanism, called Excitation Backprop. We show how such visualizations are equally informative for correct/incorrect model predictions, and with/without adoption of various training strategies.

- We propose a guided dropout regularizer for deep networks. We dropout with higher probability neurons that contribute more to decision making at training time. This approach penalizes neurons that are most relevant for model prediction. By dropping such high-saliency neurons, the network is forced to learn alternative paths in order to maintain loss minimization. We demonstrate better generalization ability, an increased utilization of the network, and a higher resilience to network compression.

- We propose `Guided Zoom`, an approach that utilizes spatial grounding to make more informed predictions at test time. `Guided Zoom` compares the evidence used to make a preliminary class prediction with evidences for class predictions seen during training. We show that this results in the refinement of model classification accuracies for three fine-grained classification datasets.

- We devise a formulation that simultaneously grounds evidence in space and time, in a single pass, using top-down saliency. We visualize the spatiotemporal cues that contribute to a deep recurrent model's classification/captioning output using the model's internal representation. Based on these spatiotemporal cues, we are able to localize segments within a video that correspond with a specific action, or phrase from a caption, without explicitly optimizing/training for these tasks.

1.3 Roadmap of Thesis

The rest of the thesis is organized as follows:

Chapter 2: Related Work

This chapter presents related works for improving deep models for classification without grounding, improving deep model classification at training time using spatial grounding, improving deep model classification at test time using spatial grounding, and extending grounding to become spatiotemporal for recurrent neural networks.

Chapter 3:

This chapter describes how the image modality can be utilized to boost the performance of the video modality for the task of classifying human actions. This is an example of improving model performance without grounding. We analyze the effect of adding web images to the training data of deep convolutional neural networks that are trained to perform action recognition from video frames, and demonstrate the complementarity of the two domains. We also demonstrate that due to the rich subject and clothing variation in n web images compared to that of n video frames, web action images can be used to replace millions of video frames in training time, maintaining performance.

Chapter 4:

This chapter gives a review of Excitation Backprop, a top-down spatial saliency approach for visual grounding. First, the assumptions, formulation, and contrastive variant of Excitation Backprop are discussed. Then, applications of the approach for model interpretation, data annotation, and domain analysis are presented. Such applications range from facial expression recognition, to medical imaging, to visualizing domain shifts.

Chapter 5:

This chapter describes our approach to utilizing model grounding for improving a model's classification ability at training time. We argue for a different dropout scheme that is guided in the way it selects neurons to be dropped, and is biased to drop neurons that contribute most to a prediction using a higher probability. Grounding is used to select such high-contribution neurons. We demonstrate that the deliberate damaging of such highly excited paths forces a network to learn alternative paths exhibiting plasticity-like behavior. We then quantitatively evaluate how this proposed regularization scheme improves generalization, increases network utilization, and makes models more robust to network compression.

Chapter 6:

In this chapter, we devise a methodology that utilizes explicit spatial grounding to refine a model's prediction at test time. Our refinement module selects one of the top-k predictions of a model based on which has the most consistent (evidence, prediction) pair with respect to (evidence, prediction) pairs of a reference pool. The reference pool is populated by evidence obtained using various spatial grounding techniques for correctly classified training data.

Chapter 7:

In this chapter, we extend spatial saliency to the temporal dimension. We present our formulation that models the top-down attention mechanism of generic CNN-RNN models to produce interpretable and useful task-relevant spatiotemporal saliency maps which enable us to visualize how recurrent models ground their decisions in images and videos. We demonstrate that such saliency maps can be used for action/caption localization in videos without explicit supervision.

Chapter 8: Conclusions and Future Work

In this chapter we summarize our contributions and discuss their strengths, limitations, and possible future directions.

1.4 List of Related Papers

Material for this thesis is based on the following papers:

1. **S.A. Bargal***, A. Zunino*, V. Petsiuk, J. Zhang, K. Saenko, V. Murino, S. Sclaroff. "Guided Zoom: Questioning Network Evidence for Fine-grained Classification." *In Submission. arXiv preprint: 1812.02626*, 2018.

2. A. Zunino*, **S.A. Bargal***, P. Morerio, J. Zhang, S. Sclaroff, V. Murino. "Excitation Dropout: Encouraging Plasticity in Deep Neural Networks." *In Submission. arXiv preprint: 1805.09092*, 2018.

3. **S.A. Bargal***, A. Zunino*, D.Kim, J. Zhang, V. Murino, S. Sclaroff. "Excitation Backprop for RNNs." In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition* (CVPR), 2018.

4. M. Monfort, B. Zhou, **S.A. Bargal**, A. Andonian, T. Yan, K. Ramakrishnan, L. Brown, Q. Fan, D. Gutfruend, C. Vondrick, A. Oliva. "Moments in Time Dataset: one million videos for event understanding." *In Submission. arXiv preprint: 1805.09092*, 2018.

5. J. Zhang, **S.A. Bargal**, Z. Lin, J. Brandt, X. Shen, S. Sclaroff. "Top-down Neural Attention by Excitation Backprop." *International Journal of Computer Vision* (IJCV), 2017.

6. S. Ma, **S.A. Bargal**, J. Zhang, L. Sigal, S. Sclaroff. "Do Less and Achieve More: Training CNNs for Action Recognition Utilizing Action Images from the Web." *The Journal of the Pattern Recognition Society* (PR), 2017.

7. **S.A. Bargal**, E. Barsoum, C. Canton, C. Zhang. "Emotion Recognition in the Wild from Videos using Images." In *Proc. of the International Conference on Multimodal Interaction* (ICMI), 2016.

Chapter 2

Related Work

In this chapter, we review related work for each of the following chapters.

2.1 Improving Model Classification for Action Recognition

Action recognition is an important model classification problem for which a large number of methods have been proposed [133]. Among these, due to promising performance on realistic videos including web videos and movies, bag-of-words approaches that employ expertly-designed local space-time features have been widely used. Some representative works include space-time interest points [68] and dense trajectories [123]. Advanced feature encoding methods, *e.g.* Fisher vector encoding [92], can be used to further improve the performance of such methods [124]. Besides bag-of-words approaches, other works make an effort to explicitly model the space-time structures of human actions [98, 127, 131] by using, for example, HCRFs and MRFs.

CNN models learn discriminative visual features at different granularities, directly from data, which may be advantageous in large-scale problems. CNN models may implicitly capture higher-level structural patterns in the features learned at the last layers of the CNN model. In addition, CNN features may also be used within structured models like HCRFs and MRFs to further improve performance.

Some recent works propose the use of CNN models for action recognition in videos

[47, 55, 84, 108]. Ji *et al.* [47] and Tran *et al.* [116] use 3D convolution filters within a CNN model to learn space-time features. Karpathy *et al.* [55] construct a video dataset of millions of videos for training CNNs and also evaluate different temporal fusion approaches. Simonyan and Zisserman [108] use two separate CNN streams: one CNN is trained to model spatial patterns in individual video frames and the other CNN is trained to model the temporal patterns of actions, based on stacks of optical flow. Wang *et al.* [128] extract trajectory-pooled deep-convolutional descriptors (TDD) from convolutional feature maps of trained two-stream ConvNets and improved trajectories to pool convolutional features centered at the trajectory. Our approach could further improve the spatial stream of the two-stream ConvNet, and therefore the TDD of Wang *et al.*. Lan *et al.* [67] introduce Multi-skIp Feature Stacking (MIFS) that utilizes multiple time skips to mimic multiple time-scales, stacking features of different frequencies, which are then Fisher encoded. Ng *et al.* [84] use a recurrent neural network that has long short-term memory (LSTM) cells. In all of these works, the CNN models are trained only on videos. Our findings regarding the use of web action images in training may help in further improving the performance of these works.

Web action images have been used for training non-CNN models for action recognition [17, 44] and event recognition [27, 125] in videos. Ikizler-Cinbis *et al.* [44] use web action images to train linear regression classifiers for small-scale action classification tasks (5 or 8 action classes). Chen *et al.* [17] use static action images to generate synthetic samples for training SVM action classifiers and evaluate on a small test set of 78 videos comprising 5 action classes. In [27], Duan *et al.* use SVMs trained on SIFT features of web action images in their video event recognition system and evaluate on datasets with 5$\sim$6 different events. Wang *et al.* [125] exploit semantic groupings of Web images for video event recognition and evaluate on the same datasets as [27]. Sun *et al.* [114] localize actions temporally

using a domain transfer from web images. Sultani and Shah [113] leverage action proposals in web images to construct action proposals in videos for the task of action localization in videos. In contrast, our work gives the first thorough study on combining web action images with videos for training CNN models for large-scale action recognition.

2.2 Spatial Grounding

There is a rich literature about modeling the top-down influences on selective attention in the human visual system (see [7] for a review). It is hypothesized that top-down factors like knowledge, expectations and behavioral goals can affect the feature and location expectancy in visual processing [135, 117, 60, 23], and bias the competition among the neurons [100, 118, 23, 22, 12]. Our attention model is related to the Selective Tuning model of [118], which proposes a biologically inspired attention model using a top-down WTA inference process.

Various methods have been proposed for grounding a CNN classifier's prediction. [145, 155] use masking-based methods to predict salient image regions. This method slides a mask over the receptive field and uses the score/response decrease as the indicator of the importance of the masked area. Recently,[30] use a meta-learning paradigm to predict the minimally salient region by editing the image and learning from the corresponding changes to its output. In [106, 145, 111], error backpropagation based methods are used for visualizing relevant regions for a predicted class or the activation of a hidden neuron. Recently, a layer-wise relevance backpropagation method is proposed by [5] to provide a pixel-level explanation of CNNs' classification decisions. [15] propose a feedback CNN architecture for capturing the top-down attention mechanism that can successfully identify task-relevant regions. The architecture requires the addition of a binary neuron feedback layer after every ReLU layer. Neurons in the feedback layer pass dominant features to

upper layers and propagate high level semantics to lower layers to create attention maps. In [156], it is shown that replacing fully-connected layers with an average pooling layer can help generate coarse class activation maps that highlight task relevant regions.

Unlike these previous methods, Excitation Backprop is based on the WTA principle, and has an interpretable probabilistic formulation. It is also conceptually simpler than [15, 155] as ir does not require modifying a network's architecture or performing additional training. The ultimate goal of this method goes beyond visualization and explanation of a classifier's decision [145, 111, 5], as it aims to maneuver CNNs' top-down attention to generate highly discriminative attention maps for the benefits of localization.

Training CNN models for weakly supervised localization has been studied by [87, 90, 88, 95, 29, 106, 35, 11]. In [87, 29, 95], a CNN model is transformed into a fully convolutional net to perform efficient sliding window inference, and then Multiple Instance Learning (MIL) is integrated in the training process through various pooling methods over the confidence score map. Due to the large receptive field and stride of the output layer, the resultant score maps only provide very coarse location information. To overcome this issue, a variety of strategies, *e.g.* image re-scaling and shifting, have been proposed to increase the granularity of the score maps [87, 95, 94]. Image and object priors are also leveraged to improve the object localization accuracy in [90, 88, 95]. [35] perform weakly supervised localization using appearance models of previously localized (segmented) classes to select and segment a new class, thereby deriving a binary segmentation mask for each image. Compared with weakly supervised localization, the problem setting of our task is essentially different. We assume a pre-trained deep CNN model is given, which may not use any dedicated training process or model architecture for the purpose of localization. Our focus, instead, is to model the top-down attention mechanism of *generic* CNN models to produce interpretable and useful task-relevant attention maps.

2.3 Spatial Grounding for Improving Model Classification at Training Time

Dropout was first introduced by Hinton *et al.* [40] and Srivastava *et al.* [112] as a way to prevent neural units from co-adapting too much on the training data by randomly omitting subsets of neurons at each iteration of the training phase.

Some follow-up works have explored different schemes for determining how much dropout is applied to neurons/weights. Wager *et al.* [120] described the dropout mechanism in terms of an adaptive regularization, establishing a connection to the AdaGrad algorithm. Inspired by information theoretic principles, Achille and Soatto [2] propose Information Dropout, a generalization dropout which can be automatically adapted to the data. Kingma *et al.* [59] showed that a relationship between dropout and Bayesian inference can be extended when the dropout rates are directly learned from the data. Kang *et al.* [52] introduces Shakeout which instead of randomly discarding units as dropout does, it randomly enhances or reverses each unit's contribution to the next layer. Wan *et al.* [121] introduced the DropConnect framework, adding dynamic sparsity on the weights of a deep model. DropConnect generalized Standard Dropout by randomly dropping the weights rather than the neuron activations in the network. Rennie *et al.* [99] proposed a time scheduling for the retaining probability for the neurons in the network. The presented adaptive regularization scheme smoothly decreased in time the number of neurons turned off during training. Recently, Morerio *et al.* [83] proposed Curriculum Dropout to adjust the dropout rate in the opposite direction, exponentially increasing unit suppression rate during training, leading to a better generalization on unseen data.

Other works focus on which neurons to drop out. Dropout is usually applied to fully-connected layers of a deep network. Conversely, Wu and Gu [136] studied the effect of dropout in convolutional and pooling layers. The selection of neurons to drop depends on

the layer where they reside. In contrast, we select neurons within a layer based on their contribution. Wang and Manning [129] demonstrate that sampling neurons from a Gaussian approximation gave an order of magnitude speedup and more stability during training. Li *et al.* [72] proposed to use multinomial sampling for dropout, *i.e.* keeping neurons according to a multinomial distribution with specific probabilities for different neurons. Ba and Frey [4] jointly trained a binary belief network with a neural network to regularize its hidden units by selectively setting activations to zero accordingly to their magnitude. While this takes into consideration the magnitude of the forward activations, it does not take into consideration the relationship of these activations to the ground-truth. In contrast, we drop neurons based on how they contribute to a network's decision.

We compare our results against Morerio *et al.* [83], which is the current state-of-the-art. To the best of our knowledge, we are the first to probabilistically select neurons to dropout based on their task-relevance.

2.4 Spatial Grounding for Improving Model Classification at Test Time

Fine-grained classification is an important model classification problem for which a large number of approaches have been proposed. The key module in fine-grained classification is finding discriminative parts. Some approaches use supervision to find such discriminative features, *i.e.* use annotation for whole object and/or for semantic parts. Zhang *et al.* [149] train part models such that the head/body can be compared, however this requires a lot of annotation of parts. Krause *et al.* [62] use whole annotations and no part annotations. Branson *et al.* [13] normalize pose of object parts before computing a deep representation for them. Zhang *et al.* [146] introduce part-abstraction layers in the deep classification model, enabling weight sharing between the two tasks. Huang *et al.* [41] introduce a part-stacked CNN which encodes part and whole object cues in parallel based on supervised

part localization. Wang *et al.* [130] retrieve neighboring images from the dataset, those having similar object pose, and automatically mine discriminative triplets of patches with geometric constraints as the image representation. Deng *et al.* [21] include humans in the loop to help select discriminative features. Subsequent work of Krause *et al.* [63] does not use whole or part annotations, but augments fine-grained datasets by collecting web images and experimenting with filtered and unfiltered versions of them. Wang *et al.* [122] use the ontology tree to obtain hierarchical multi-granularity labels. In contrast to such approaches, we do not require any whole or part annotations at train or test time and do not use additional data or hierarchical labels.

Other approaches are weakly supervised. Such approaches only require an image label, and our approach lies in this category. Lin *et al.* [73] demonstrate the applicability of a bilinear CNN model in the fine-grained classification task. Sun *et al.* [115] implement an attention module that learn to localize different parts and a correlation module to coherently enforce correlations among different parts in training. Fu *et al.* [31] learn where to focus by recurrently zooming into one location from coarse to fine using a recurrent attention CNN. In contrast, we are able to zoom into multiple image locations. Zhang *et al.* [151] use convolutional filters as part detectors since the responses of distinctive filters usually focus on consistent parts. Zhao *et al.* [152] use a recurrent soft attention mechanism that focuses on different parts of the image at every time step. This work enforces a constraint to minimize the overlap of attention maps used in adjacent time steps to increase the diversity of part selection. Zheng *et al.* [154] implement a multiple attention convolutional neural network with a final fully-connected layer combining the softmax for each part with one classification loss function. Cui *et al.* [19] introduce a kernel pooling scheme and also demonstrate benefit to the fine-grained classification task. Jaderberg *et al.* [45] introduce spatial transformers for convolutional neural networks which results in models which learn

invariance to translation, scale, rotation and more generic warping, showing improvement for the task of fine-grained classification.

In contrast, our approach assesses whether the network evidence used to make a prediction is reasonable, *i.e.* if it is coherent with the evidence of correctly classified training examples of the same class. We use multiple salient regions eliminating error propagation from incorrect initial saliency localization, and implicitly enforce part-label correlations enabling the model to make more informed predictions at test time.

2.5 Spatiotemporal Grounding

Spatiotemporal grounding is grounding the evidence of a model's prediction both in space and time. Several works in the literature give more insight into CNN model predictions, *i.e.* , the *evidence* behind deep model predictions in space. Such approaches are mainly devised for image understanding and can identify the importance of class-specific image regions by means of saliency maps in a weakly-supervised way.

Spatial Grounding. Ribeiro *et al.* [101] explained classification predictions with applications on images. Fong *et al.* [30] addressed spatial grounding in images by exhaustively perturbing image regions. Guided Backpropagation [111] and Deconvolution [145, 106] used different variants of the standard backpropagation error and visualized salient parts at the image pixel level. In particular, starting from a high-level feature map, [145] inverted the data flow inside a CNN, from neuron activations in higher layers down to the image level. Guided Backpropagation [111] introduced an additional guidance signal to standard backpropagation preventing backward flow of negative gradients. Simonyan *et al.* [106] directly computed the gradient of the class score with respect to the image pixel to find the spatial cues that help the class predictions in a CNN. CAM [156] removed the last fully connected layer of a CNN and exploited a weighted sum of the last convolutional feature

maps to obtain the class activation maps. Zhang *et al.* [148] generated class activation maps from any CNN architecture that uses non-linearities producing non-negative activations. Oquab *et al.* [86] used mid-level CNN outputs on overlapping patches, requiring multiple passes through the network.

Spatiotemporal Grounding. Weakly-supervised visual saliency is much less explored for temporal architectures. Karpathy *et al.* [54] visualized interpretable LSTM cells that keep track of long-range dependencies such as line lengths, quotes, and brackets in a character-based model. Li *et al.* [70] visualized a unit's salience for NLP. Selvaraju *et al.* [103] qualitatively present grounding for captioning and visual question answering in images using an RNN. Ramanishka *et al.* [97] explored visual saliency guided by captions in an encoder-decoder model. In contrast, our approach models the top-down attention mechanism of CNN-RNN models to produce interpretable and useful task-relevant spatiotemporal saliency maps that can be used for action/caption localization in videos.

2.6 Discussion

In this chapter we present related work for this thesis. In Section 2.1 we review different approaches for action recognition in videos. We also review various work that utilizes web action images to aid several vision tasks including action recognition. Such approaches improve model classification, but lack the capability of explaining classification decisions made by the model. Section 2.2 introduces approaches used to visually ground why deep models make certain predictions for spatial visual data, *i.e.* when the input to the model is an image. We demonstrate various applications on how grounding techniques can differentiate between domains and how they can be used to highlight the shift of focus using different training strategies. Section 2.3 reviews variants of a regularization scheme that is widely used in deep models, and introduces our variant that performs regularization in a manner

that is guided by spatial grounding. Our proposed scheme results in improved network generalization on unseen test data, increased network utilization, and increased robustness to network compression. Section 2.4 reviews approaches to fine-grained classification of images and introduces how we approach the problem using spatial grounding. Our approach refines the classification accuracy for fine-grained datasets at test time by evaluating how coherent a grounded evidence of a prediction is compared to (evidence, prediction) pairs of correctly classified training examples. Section 2.5 demonstrates how spatiotemporal grounding for visual data is much less explored compared to spatial grounding, *i.e.* when the input to the model is a video. We present a formulation for top-down attention in recurrent neural networks for spatiotemporal grounding. In the next chapters, we will present our contributions to the areas of the literature summarized here.

Chapter 3

Do Less and Achieve More: Training CNNs for Action Recognition Utilizing Action Images from the Web

Deep neural network models are state-of-the-art for many Computer Vision tasks. Recent works [55, 107] show that deep convolutional neural networks (CNNs) are promising for action recognition in videos. However, CNN models typically have millions of parameters [16, 66, 108], and usually large amounts of training data are needed to avoid overfitting. For this purpose, work is underway to construct datasets consisting of millions of videos [55]. However, the collection, pre-processing, and annotation of such datasets can require a lot of human effort. Moreover, storing and training on such large amounts of data can consume substantial computational resources.

In contrast, collecting and processing images from the Web is much easier. For example, one may need to look through all, or most, video frames to annotate the action, but often a single glance is enough to decide on the action in an image. Videos and web images also have complementary characteristics. A video of 100 frames may convey a complete temporal progression of an action. In contrast, 100 web action images may not capture the temporal progression, but do tend to provide more variations in terms of camera viewpoint, background, body part visibility, clothing, *etc.* Moreover, videos often contain many

Figure 3.1: Sample action images from BU101. Action images on the Web often capture well-framed descriminative poses of the actions they represent. Left to right: *Hammer Throw*, *Body Weight Squats*, *Jumping Jack*, *Basketball*, *Tai Chi*, *Cricket Shot*, *Lunges*, *Still Rings*. Utilizing web action images in training CNNs, for all these action classes, results in more than 10% absolute increase in recognition accuracy in videos compared to CNNs trained only on video frames (see Fig. 3.3).

redundant and uninformative frames, *e.g.*, standing postures, whereas action images tend to focus on discriminative portions of the action (Fig. 3.1). This property can further focus the learning, making action images inherently more valuable.

In summary, two intuitions emerge about why web action images may be useful in training CNNs for action classification of videos:

- *Complementarity*: Action images may complement training videos when video data is scarce, particularly since images may be easier to collect and process.
- *Efficiency*: Web action images usually contain discriminative poses of the actions, making them intrinsically higher-quality training data compared to video frames, which may be redundant or contain less relevant poses.

However, it is not enough to stop at these seemingly natural intuitions: scientific verification is necessary. In this work, we analyze and empirically evaluate these intuitions. To our best knowledge, we are the first to perform an in-depth analysis of this problem by extensive and large-scale empirical evaluation.

We start by collecting large web action image datasets. The first dataset, BU101, contains 23.8K images of 101 action classes. It is more than double the size of the largest previous action image dataset [141], both in the number of images and the number of actions. And, to the best of our knowledge, this is the first action image dataset that has

one-to-one correspondence in action classes with the large-scale action recognition video benchmark dataset, UCF101 [58]. Images of the dataset are carefully labeled and curated by human annotators; we refer to them as *filtered* images. Two other, even larger, web image datasets are also collected: BU101-unfiltered and BU203-unfiltered, which are crawled automatically by querying action class names on multiple image search engines, *e.g.* Google Image Search. The BU101-unfiltered dataset contains $\sim$0.2M images crawled by querying the 101 action class names of UCF101, and BU203-unfiltered contains $\sim$0.4M images crawled by querying the 203 activity names of ActivityNet [14]. All these datasets will be made publicly available to the research community [1].

We train CNN models of different depths and analyze the effect of adding web action images of BU101 to the training set of video frames. We also train and evaluate models with varying numbers of action images to explore the marginal gain as a function of the web image set size. We find that by combining web action images with video frames in training, a spatial CNN can achieve an accuracy of 83.5% on UCF101, which is more than a 10% absolute improvement over a spatial CNN trained only on videos [107]. When combining with motion features, we can achieve 91.1% accuracy, which is the highest result reported to-date on UCF101. We also replace videos by images to demonstrate that our performance gains are due to images providing complementary information to that available in videos, and not solely due to additional training data.

We further investigate at a larger scale, *i.e.* use many more web action images as additional training data, where these action images are simply automatically crawled and without further annotation. We compare the performance of using BU101 and BU101-unfiltered images on UCF101. Using BU101-unfiltered we obtain similar performance to that obtained using BU101, even though collecting BU101-unfiltered requires much less

[1] http://www.cs.bu.edu/groups/ivc/BU-action/

human labor. We also obtain comparable performance when replacing half the training videos in ActivityNet (which correspond to 16.2M frames) by $\sim$ 400K images of BU203-unfiltered.

We then delve deeper and examine one major mechanism which may deliver the benefits of web action images in training the CNN models. We bring to light an artifact of finetuning a pre-trained CNN: *conservative filters* – CNN filters that undergo small changes during fine-tuning and make little, if any, contribution to the target task. These conservative filters reduce the number of effective parameters in the CNN model and are potentially harmful to the modeling capacity of the CNN. We illustrate that, by using web action images as additional training data, the number of conservative filters is greatly reduced, *e.g.* by an order of magnitude. This enables re-targeting more filters of the pre-trained CNN to visual patterns of the new task, *i.e.* action recognition.

In summary, our **contributions** are:

- We collect three large web action image datasets: BU101, BU101-unfiltered and BU203-unfiltered. These datasets are in one-to-one correspondence with the actions in the UCF101 or ActivityNet benchmark datasets.

- By extensive experimental evaluation, we verify the intuition that web action images are complementary to video training data. This complementarity appears to be insensitive to the CNN depth and is evident in many kinds of actions. Benefits are observed even when only a few filtered images are used in training and the benefits grow with number of web images.

- We illustrate that both filtered and unfiltered web action images are complementary to video training data. This points to an approach that requires little human annotation labor and is especially useful for large-scale problems.

- We show that using web action images can boost the efficiency of CNN training. With the same number of training samples, the trained model can achieve significantly higher recognition performance if half of the samples are web images. Moreover, to achieve the same recognition performance, we can greatly reduce the number of training videos and use unfiltered web action images instead.

- We provide insight into an artifact of finetuning a pre-trained CNN model for a new task: *conservative filters*. We show that, in our action recognition task, by using web action images as additional training data, the number of conservative filters can be significantly reduced. This reveals an underlying mechanism that brings in the benefits of web action images in the CNN model finetuning.

3.1 Web Action Image Datasets

To study the usefulness of web action images for learning better CNN models for action recognition, we collect action images that correspond with the 101 action classes in the UCF101 video dataset and the 203 activities in the ActivityNet dataset (version 1.1). This leads to 3 large image datasets: BU101, BU101-unfiltered and BU203-unfiltered. All three datasets will be made publicly available for research.

We collect images by crawling the web using the action class names of UCF101 or ActivityNet as queries on image search engines (Google Image Search, Flickr, etc.). Some queries are augmented by the words *exercise*, *train* and *play* when appropriate, *e.g.* , *juggling balls* to *play juggling balls*. BU101-unfiltered and BU203-unfiltered, containing 204K images and 387K images respectively, are simply compiled by this crawling procedure using action (activity) names of UCF101 (ActivityNet), without any further human annotation. We conducted a study to ensure that our curated datasets do not include frames of the

Dataset	# Actions	# Images	Clutter?	Poses vary?	Visibility varies?
Gupta [36]	6	300	Small	Small	No
Ikizler [43]	5	1727	Yes	Yes	Yes
VOC2012 [28]	11	4588	Yes	Yes	Yes
PPMI [140]	24	4800	Yes	Yes	No
Standford40 [141]	40	9532	Yes	Yes	Yes
BU101	**101**	**23800**	**Yes**	**Yes**	**Yes**

Table 3.1: Comparison of BU101 with existing action image datasets. *Visibility varies?* refers to variance in the partial visibility of the human bodies.

corresponding video datasets (Section 3.3).

In the following text, we focus our discussion on the dataset BU101. For BU101, we inspect each crawled image and remove images that do not contain the action or are cartoons or drawings. We also include 2769 images of relevant actions from the Standford40 dataset [141]. The resulting dataset comprises 23.8K images. Because the images are automatically collected, and then filtered for irrelevant ones, the number of images per category varies. Each class has at least 100 images and most classes have 150-300 images.

Table 3.1 compares existing action image datasets with our new dataset, BU101. Both in the number of images and the number of actions, our dataset exceeds double the scale of existing datasets. More importantly, to the best of our knowledge, this is the first action image dataset that has one-to-one action class correspondence with a large-scale action recognition benchmark video dataset. We believe that our dataset will enable further study of the relationship between action recognition in videos and in still images.

UCF101 action classes are divided into five types: *Human-Object Interaction*, *Body-Motion Only*, *Human-Human Interaction*, *Playing Musical Instruments*, and *Sports* [110]. Fig. 3.2 shows sample images in BU101 for five action classes, one in each of the five action types.

These action images collected from the Web are originally produced in a variety of settings, such as amateur *vs.* professional photos, artistic *vs.* educational *vs.* commercial

Figure 3.2: Sample images from BU101. Each row shows images of one action. Top to bottom: *Hula Hoop, Jumping Jack, Salsa Spin, Drumming, Frisbee Catch*. Variations in background, camera viewpoint and body part visibility are common in web images of the same action.

photos, etc. For images collected in each action category, wide variation can exist in viewpoint, lighting, human pose, body part visibility, and background clutter. For example, commercial photos may have clear backgrounds while backgrounds of amateur photos may contain much more clutter. Such variance also differs for different types of actions. For example, for *Sports*, there is significant variance in body pose among images that capture different phases of the actions, whereas body pose variance is minimal in images of *Playing Musical Instruments*.

Many of the collected action images significantly differ from video frames in camera viewpoint, lighting, human pose, and background. One interesting thing to notice is that action images often capture *defining poses* of an action that are highly discriminative, *e.g.* standing with both hands over head and legs spread in *jumping jack* (Fig. 3.2, row 2). In contrast, videos may have many frames containing poses that are common to many actions, *e.g.* in *jumping jack* the upright standing pose with hands down. Also, n images

will have more unique content than n video frames, for example more clothing variation. Clearly there exists a compromise between temporal information available in videos and discriminative poses and variety of unique content in images.

3.2 Training CNNs with Web Action Images

Spatial CNNs trained on single video frames for action recognition are explored in [107]. Karpathy *et al.* [55] observe that spatio-temporal networks show similar performance compared to spatial models. A spatial CNN effectively classifies actions in individual video frames, and action classification for a video is accomplished via fusion of the spatial CNN's outputs over multiple frames, *e.g.* via voting or SVM. Because the spatial CNN is trained on single video frames, its parameters can be learned by fine-tuning of a CNN that was trained for a different task, *e.g.*, using a CNN that is pre-trained on ImageNet [20]. The fine-tuning approach is especially beneficial in training a CNN model for action classification in videos, since we often only have limited training samples; given the large number of parameters in a CNN, initializing the parameters to random values leads to overfitting and inferior performance as shown in [107]. In this work, we study improving the spatial CNN for action recognition using web action images as training data. This is then combined with motion features via state-of-the-art techniques.

In our experiments and analysis, we explore the following key questions:

- Is it beneficial to train CNNs with web action images in addition to video frames and, if so, which action classes benefit most?
- How do different CNN architectures, in particular ones with different depths, perform when web action images are used as additional training data?
- How do the performance gains change when more web action images are used in

training the CNN?

- Are performance gains solely due to additional training data or also due to a single image being more informative than a randomly sampled video frame?
- Can we make the procedure of leveraging web images scalable by using crawled (unfiltered) web images rather than manually filtered ones?

We experiment on three CNN architectures: M2048 [16], VGG16, and VGG19 [108]. To avoid cluttering the discussion, implementation details are provided later in Sec. 3.5.

Is adding web images beneficial? Significant performance gains are achieved when we train spatial CNNs using BU101 as auxiliary training data (see Table 3.2). For example, with the VGG19 CNN architecture, 5.7% absolute improvement in mean accuracy is achieved.

Most encouragingly, such improvements are easy to implement, without the need to introduce additional complexity to the CNN architecture and/or requiring significantly longer training time.

We further analyze which classes improve the most. Fig. 3.3 shows the 25 action classes for which the largest improvement in accuracy is achieved with the three different CNN architectures on UCF101 split1. The 25 action classes of top average accuracy improvement over all three tested architectures are also shown (rightmost column), all of which have no less than 10% absolute increase in accuracy and 10 classes have more than 20% absolute improvement. Some action classes are consistently improved irrespective of the CNN architecture used, such as *push ups*, *YoYo*, *handstand walking*, *brushing teeth*, *jumping jack*, etc. This suggests that utilizing web action images in CNN training is widely applicable.

While classification accuracy improvements in actions that are relatively *stationary* such as *Playing Daf* and *Brushing Teeth* are somewhat expected, it is interesting to see that improvements for actions of fast body motion such as *Jumping Jack* and *Body Weight*

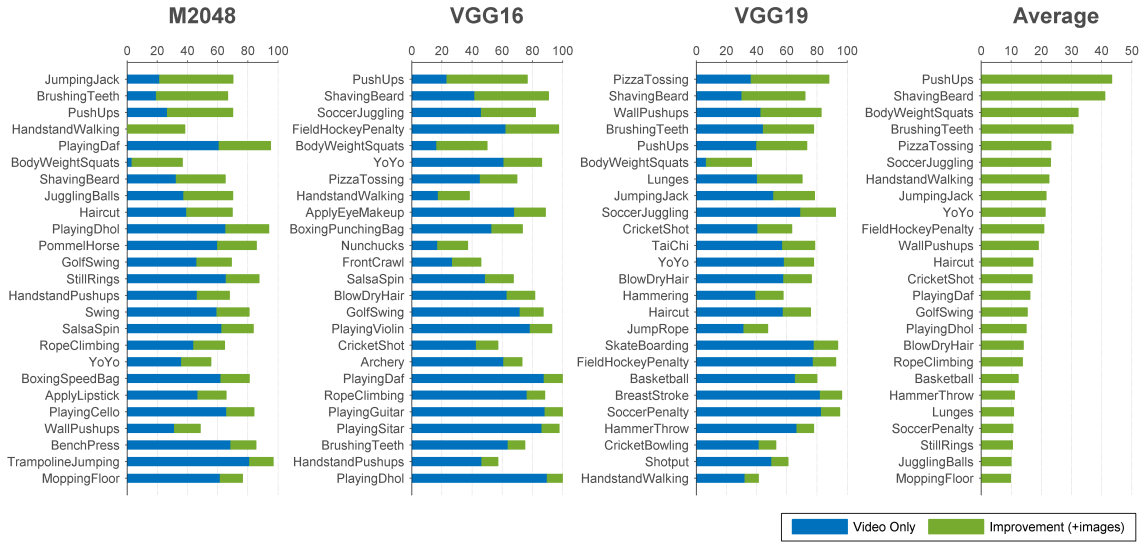


Figure 3.3: The 25 action classes with the largest accuracy improvement in the three CNN architectures as well as on average over the three architectures. The blue bars show the accuracy of CNN models trained only on videos. The green bars show the absolute increase in accuracy of CNN models trained using both web action images and training videos.

Squats are also significant.

Are images benefitial irrespective of CNN depth? While there are numerous ways that CNN architectures may differ from each other, here we focus on one of the most important factors. We evaluate the performance changes for CNNs of different depths when BU101 is used in addition to video frames in training. We train spatial CNNs of three depths: 7 layers (M2048), 16 layers (VGG16) and 19 layers (VGG19). These are the prototypical choices of CNN depths in recent works [16, 66, 74, 107, 108].

Table 3.2 shows the mean accuracy of the three CNN models trained *with* and *without* BU101 on UCF101 split1. Using web action images in training leads to a consistent 5% $\sim$ 9% absolute improvement for all three architectures of different depths. This shows the usefulness of web action images and suggests a wide applicability of this approach. Furthermore, our results in action recognition confirm [108]'s observation that deeper CNNs of 16-19 layers significantly outperform the shallower 7-layer architecture. However,

Model	# Layers	# Parameters (in Millions)	Accuracy video only	Accuracy video + images
M2048	7	91	66.1%	**75.2%**
VGG16	16	138	77.8%	**83.5%**
VGG19	19	144	78.8%	**83.5%**

Table 3.2: Accuracy on UCF101 split1 using three different CNN architectures.

the margin of performance gain diminishes when we increase the depth from 16 to 19.

Does adding more web images improve accuracy? We further explore how, for the same CNN architecture, the number of web action images used as additional training data can influence the classification accuracy of the resulting CNN model. We sample $1/10$, $1/5$, $1/3$ and $2/3$ of the images of each action in our dataset, and for each sampled set we train the spatial CNN by fine-tuning VGG16 using both the training videos and sampled action images from BU101. For each sample size, we repeat the experiment three times, each with a different randomly sampled set of web action images. The evaluation is performed on UCF101 split1.

Fig. 3.4 summarizes the results of this experiment. The increase in classification accuracy is most significant at the beginning of the curve, *i.e.* when a few thousand web action images are used in training. This increase continues as more web action images are used, even though the increase becomes slower. Firstly, this indicates that using web action images in training can make a significant difference in performance by providing additional supervision to that provided by video frames. Secondly, it indicates that it is good practice to collect a moderate number of web action images for each action as a cost-effective way to boost model performance (*e.g.*, $100 \sim 300$ images per action for a dataset of the same scale as UCF101).

Do web images complement video frames? Although augmenting with images is more efficient than augmenting with videos, we further investigate whether the achieved perfor-

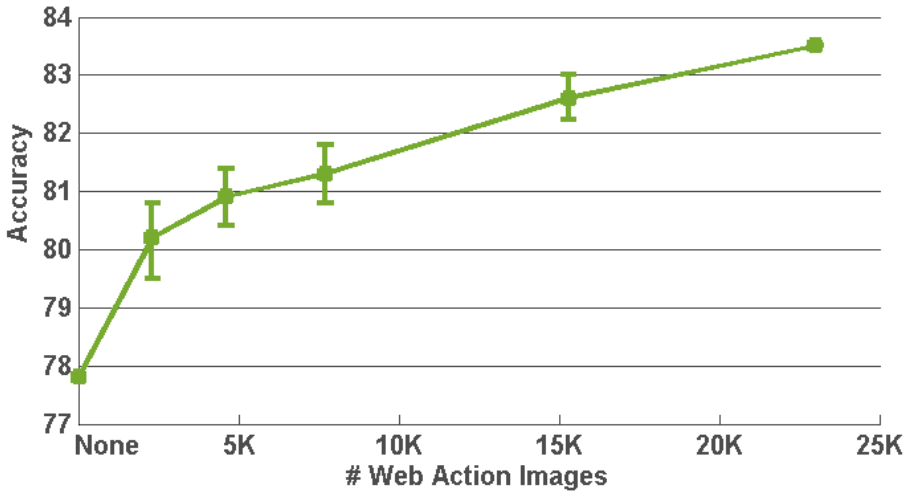


Figure 3.4: Performance of the spatial CNNs (VGG16) trained on UCF101 split1 using different numbers of web action images of BU101 as additional training data.

mance gains are solely due to additional training data or whether a web image provides more information to the learning algorithm than a video frame. This is done by replacing video frames by web images of BU101, keeping the total number of training samples constant. For each sample size, we repeat the experiment three times, each with a different randomly sampled set of web action images. The evaluation is performed on UCF101 split1 and a VGG16 model.

Fig. 3.5 summarizes the results of this experiment. A consistent improvement in performance is achieved when half the video frames are replaced by web images. The number of training samples (images and video frames) required to obtain the maximum accuracy presented in Fig. 3.4 is much less (50K *vs.* 230K). This suggests that images are augmenting the information learnt by the classifier. We posit that discriminative poses in action images may provide implicit supervision, in training, to help learn better discriminative models for classification.

Can this be made scalable? While we have demonstrated the ability to collect a filtered dataset for our desired classes, this is not scalable. Given a different dataset having the

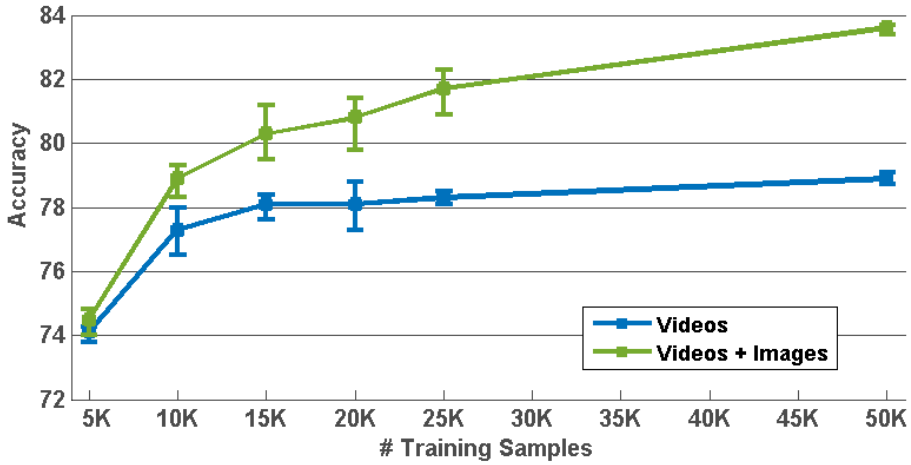


Figure 3.5: Performance of the spatial CNNs (VGG16) trained on UCF101 split1 using video frames only and replacing 50% of the video frames by web images.

same order of magnitude as UCF101 we would have to manually label a dataset for its classes. Given an even larger dataset with more classes and more samples per class, this becomes very cumbersome although still better than collecting videos. We now investigate the possibility of using crawled (unfiltered) web images for the same purpose, utilizing BU101-unfiltered.

Table 3.3 summarizes the results of this experiment. The performance of using unfiltered images approaches that of manually filtered images, but the number of web images utilized is much larger. We further investigate whether *all* the crawled unfiltered images are required to obtain such performance. We do this by randomly selecting one quarter (65.5K) of the 204K unfiltered web images. We select 3 random samples and report the average result in Table 3.3. Three quarters of the images only contribute with an additional accuracy of 1%; this is consistent with Fig. 3.4 observations.

Having demonstrated the feasibility of using crawled web images, we now apply this to an even larger-scale dataset: ActivityNet [14] using BU203-unfiltered. ActivityNet contains more classes (203) and more samples per class than UCF101. ActivityNet classes are more diverse; they belong to the categories: *Personal Care, Eating and Drinking, Household,*

Image Type	# Images	Accuracy (%)
Manually filtered	23.8K	83.5
Unfiltered (all)	204K	83.1
Unfiltered (rand select)	65.5K	82.1*

Table 3.3: Accuracy on UCF101 split1 using spatial CNN (VGG16) of manually filtered and unfiltered web images. The symbol * inidicates an average over three random sample sets.

Caring and Helping, Working, Socializing and Leisure, and Sports and Exercises. "ActivityNet provides samples from 203 activity classes with an average of 137 untrimmed videos per class and 1.41 activity instances per video, for a total of 849 video hours." [14] Mostly, videos have a duration between 5 and 10 minutes and have a 30 FPS frame rate. About 50% of the videos are in HD resolution. Results on ActivityNet are reported in Sec. 3.5.

3.3 Image-Frame Overlap Study

In this section we conduct a study to ensure that the images composing our web action image datasets do not overlap with the video frames of the UCF101 and ActivityNet datasets. This is conducted to make certain that the web images are solely contributing due to their additional beneficial modality, and not because they contain frames of the tested datasets: UCF101 and ActivityNet.

We compare the color histograms of the curated web image datasets and the video frames. A histogram is computed using the concatenation of the R-G-B histograms, each having 256 bins, leading to an overall histogram $\in \mathbb{Z}^{768}$. For every web image, the video frame having the closest histogram, using the Manhattan distance, is saved side-by-side with the web image being examined. Every resulting pair of images is manually labeled: "same" or "different." The label "same" is given if the pair of images are depicting the same scene, even if translation, rotation, zoom in, or zoom out are present, otherwise, the label

Figure 3.6: Overlap between the web images of BU101-unfiltered and videos frames of UCF101. In each presented pair: (left) the web image and (right) the closest video frame as per color histogram. Only these two images, out of 204K in the web image dataset, matched video frames in UCF101.

"different" is given.

For UCF101, every one of the BU101-unfiltered 204K web image histograms is compared against every one of the sampled video frame histograms that belong to the same class. Through this detailed examination process, only two overlaps were found between the video frames and the web images used for augmenting the training set. These two images are presented in Figure 3.6.

For ActivityNet, we randomly sampled 200K of the original 387K of BU203-unfiltered and performed the same comparison. Every web image is compared to all sampled frames of all validation videos. Through this detailed examination process, only one overlap was found between the video frames and the web images used for augmenting the training set. This overlap is a blank (black) image.

3.4 Conservative Filters

For tasks that have limited training data, training a deep CNN by *fine-tuning* from a CNN pre-trained on a large-scale dataset (but for a different task) is an important and popular technique [32, 107]. In such an approach, most parameters (usually all but the final layer) of the target model are initialized to the parameter values of the pre-trained model. This

initialization usually works significantly better than random parameter initialization. In this work we train the spatial CNNs for target task (action recognition) by fine-tuning models that were pre-trained on ImageNet for a different task (object recognition). Despite the benefits of the fine-tuning technique, we seek to delve deeper and provide insight to its downside–what we call **conservative filters**: filters whose parameters do not change significantly during fine-tuning and whose activation is relatively higher for the pre-trained task *vs.* the target task after fine-tuning. These conservative filters reduce the effective number of parameters in the CNN for the target task and may be harmful to the CNN's modeling capacity for the target task. This study of conservative filters enables us to examine what is happening during training and quantify the impact of including web images in fine-tuning.

We investigate conservative filters in the VGG16 model that is fine-tuned using only video frames of the training videos of UCF101 split1. For the investigation, we compile an image pool containing 50K images from ImageNet and 55K video frames (*i.e.* 0.5K video frames per action class). All images are re-sized to 224×224, which is the input size of the CNNs we fine-tuned. Let $\mathbf{w}_n^k$ represent the parameter values of the kth convolution filter in the nth convolutional layer in the pre-trained CNN model and $\hat{\mathbf{w}}_n^k$ the parameter values after fine-tuning. For a filter $\hat{\mathbf{w}}_n^k$ in the fine-tuned CNN, we find its maximum activation for each image in the pool. For example, for a filter in the 13th convolutional layer (*conv5-3*), its receptive field is 211×211, so for each image in the pool we find an image patch of 211×211 that causes maximum activation. For the kth filter in the nth layer, we sort the images in descending order of their maximum activation value for this filter, and then compute the percentage of the images among the top 100 that are video frames (denoted as α_n^k). We also compare the filter's parameter change against the original pre-trained model, measured by $\Delta_n^k = \|\hat{\mathbf{w}}_n^k - \mathbf{w}_n^k\|$. We then look for conservative filters that have both small

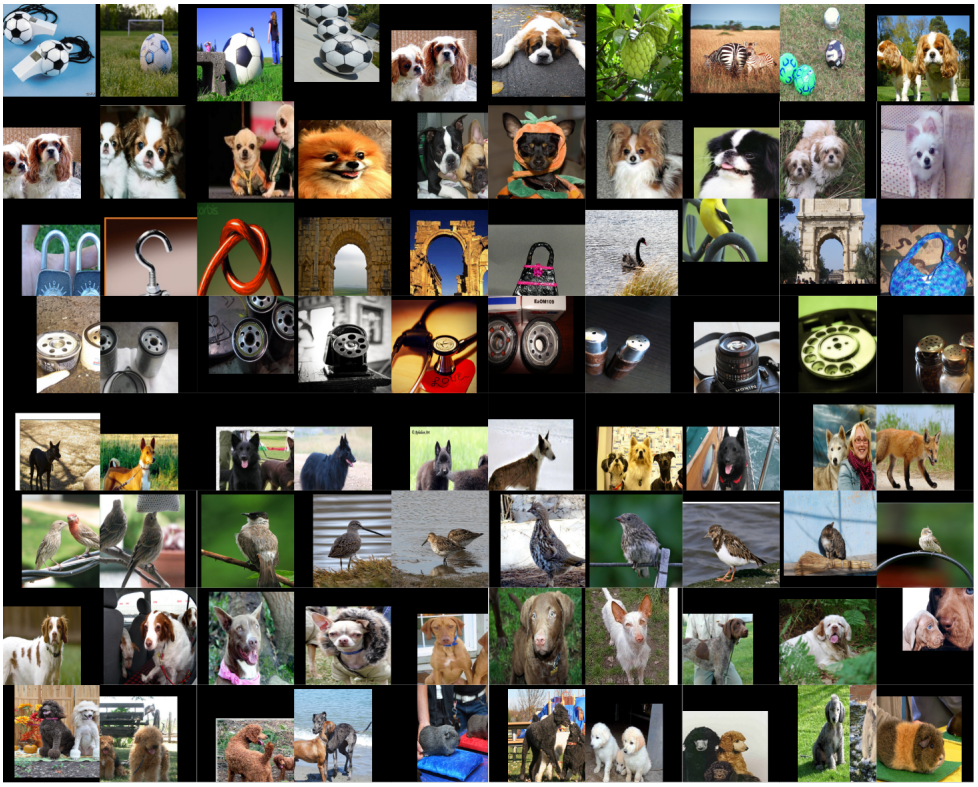


Figure 3.7: Top activations for 8 example conservative filters. Each row shows top 10 activations of one conservative filter in the last convolutional layer (conv5-3) of the VGG16 model that is fine-tuned using only video frames of the training videos in UCF101 split1.

α and Δ, *i.e.* filters that stay almost the same during fine-tuning and whose top activations are mostly on ImageNet images. These filters have relatively small activation on video frames.

Fig. 3.7 depicts the top 10 activations of the 8 filters from *conv5-3* that result from the intersection of the 30 filters of the least α and the 30 filters of the least Δ. Even though our image pool has almost equal numbers of images from ImageNet and UCF101 video frames, all these top activations come from ImageNet and correspond to dogs, objects, architecture *etc.*, which are indeed rare in UCF101.

We believe that conservative filters essentially take up parameters in the CNN that otherwise could have been used for learning visual patterns in the target task. Fine-tuning

		$\Delta < 0.02$	$\Delta < 0.03$	$\Delta < 0.04$	$\Delta < 0.05$	$\Delta < 0.06$
conv5-2	*video only*	0	2	10	30	87
	video + image	0	0	0	2	8
conv5-3	*video only*	0	13	33	82	123
	video + image	0	1	4	20	38
fc6	*video only*	12	84	212	399	636
	video + image	1	24	63	121	185

Table 3.4: Using web action images can significantly reduce the number of conservative filters. For the VGG16 model finetuned with only training video frames of UCF101 split1 (noted as *video only* in the table) and the VGG16 model finetuned with both video frames and images of BU101 (noted as *video + image*), we compute the number of filters which satisfy: 1) $\alpha < 0.1$, *i.e.* only less than 10% of top 100 activation images in our image pool are video frames; 2) Δ is less than a small value (given in the table), *i.e.* doesn't change much during finetuning. Notice the large reduction in the numbers of such filters when using web action images as additional training data.

from a pre-trained model provides good initial values for some of the parameters, but may set some parameters to bad local minima with respect to the target task. However, this situation can be significantly improved by utilizing web action images as additional training data.

To test this, we compare the number of conservative filters in the *conv5-2*, *conv5-3* and *fc6* layers in the VGG16 model fine-tuned only with training video frames of UCF101 split1 and the VGG16 model fine-tuned with both these training video frames and the web action images in BU101. Table 3.4 shows this comparison: for each model and layer, we list the number of filters whose $\alpha < 0.1$, *i.e.* less than 10% of the top 100 activation images in our image pool are video frames, and whose Δ is small, *e.g.* less than 0.05. We can observe a significant reduction of the number of conservative filters when fine-tuning using both video frames and web action images. For example, for layer *conv5-2*, the number of filters with $\alpha < 0.1$ and $\Delta < 0.06$ is reduced from 87 to 8 when web action images are used as additional training data, which is more than an order-of-magnitude reduction.

The observed correlation between the decrease of the conservative filters and the usage of the web images suggests that without web images, many filters are still highly activated

by the visual concepts that are common in the original domain, but are irrelevant to the target domain. Recall that the action classification accuracy of the VGG16 model fine-tuned only with video frames is 77.8%, and the VGG16 model fine-tuned with both video frames and web images of BU101 is 83.5% (Table 3.2): the absolute improvement after using web action images is 5.7%. We posit that the reduction of conservative filters may be an important reason for this performance improvement: web action images may help reduce the number of conservative filters so that their parameters could be *re-used* in learning visual patterns for the new task, *i.e.* action recognition in videos.

In typical deep CNNs [66, 108], the ReLU (Rectified-Linear Unit) layer follows the convolutional layers or fully-connected layers and introduces nonlinearity to the model. The ith output of $\mathbf{w}_n^k$ can be simply written as a dot product $x_n^{k,i} = \mathbf{w}_n^k \cdot \mathbf{x}_{n-1}^i$, where $\mathbf{x}_{n-1}^i$ is the part of the input to the nth layer that participates in the ith convolution. Now suppose there is a ReLU layer following this layer, and its ith output on the kth input channel can be denoted as $r_n^{k,i} = max(0, x_n^{k,i})$. During training, in back-propagation, the gradients of training loss with respect to $\mathbf{w}_n^k$ are

$$\frac{\partial L}{\partial \mathbf{w}_n^k} = \sum_i \frac{\partial L}{\partial x_n^{k,i}} \frac{\partial x_n^{k,i}}{\partial \mathbf{w}_n^k} = \sum_i \frac{\partial L}{\partial r_n^{k,i}} \frac{\partial r_n^{k,i}}{\partial x_n^{k,i}} \cdot \mathbf{x}_{n-1}^i. \tag{3.1}$$

Notice that $\partial r_n^{k,i} / \partial x_n^{k,i} = 0$ when $x_n^{k,i} < 0$ and otherwise equal to 1, so Equation 3.1 can be written as

$$\frac{\partial L}{\partial \mathbf{w}_n^k} = \sum_{i:\ x_n^{k,i} \geq 0} \frac{\partial L}{\partial r_n^{k,i}} \cdot \mathbf{x}_{n-1}^i. \tag{3.2}$$

Thus, $\partial L / \partial \mathbf{w}_n^k$ is determined by the non-negative convolution outputs, *i.e.* the set $X_n^{k,+} = \{x_n^{k,i} | x_n^{k,i} \geq 0,\ 1 \leq i \leq N\}$, where N is the total number of convolutions by $\mathbf{w}_n^k$ in the nth layer. In training a CNN, typically the following weight update is used in back-propagation

in the tth iteration:

$$\mathbf{v}_{n,t}^k = \mu \cdot \mathbf{v}_{n,t-1}^k - \delta \cdot \epsilon \cdot \mathbf{w}_{n,t-1}^k - \epsilon \cdot \left\langle \frac{\partial L}{\partial \mathbf{w}_n^k} \right\rangle_{D_t}, \tag{3.3}$$

$$\mathbf{w}_{n,t}^k = \mathbf{w}_{n,t-1}^k + \mathbf{v}_{n,t}^k. \tag{3.4}$$

where $\mathbf{v}_{n,t}^k$ is the momentum variable, μ is the momentum coefficient, δ is the weight decay coefficient and ϵ is the learning rate. Typical choices for μ and δ are 0.9 and 0.0005 respectively. The learning rate ϵ is usually small for fine-tuning, *e.g.* initialized to 10^{-3} and further reduced during training in our experiments. $\left\langle \frac{\partial L}{\partial \mathbf{w}_n^k} \right\rangle_{D_t}$ represents the average gradient over the training batch D_t. If $\mathbf{w}_n^k$ produces negative outputs for most samples in D_t, $\left\langle \frac{\partial L}{\partial \mathbf{w}_n^k} \right\rangle_{D_t}$ is very likely to be very small: in this situation, the update to $\mathbf{w}_n^k$ will be mainly from weight decay, which is also small due to the small μ and δ. If $X_n^{k,+}$ is empty or small most of the time in training, the difference of $\hat{\mathbf{w}}_n^k$ with $\mathbf{w}_n^k$ is likely to be small too, *i.e.* $\hat{\mathbf{w}}_n^k \approx \mathbf{w}_n^k$.

This situation is possible, especially when the training data of the target task (*target data*) differs significantly from the data used for pre-training (*source data*). Some filters in the pre-trained model may have learned some visual patterns in the source data that rarely appear in the target data, so that in fine-tuning their outputs may tend to be negative most of the time in the forward passes and receive very small updates in the backward passes, which can make them *stale* in the training. Clearly, these conservative filters will make small, if any, contributions to the target task. Also, note that each unit of a fully-connected layer can be seen as a filter with size equal to its whole input, so the discussion above also holds for the fully-connected layers.

Our observation and our analysis of the ReLU units suggest that the domain change between images and videos, together with the low diversity of video frames, may obstruct the adaption process of some high-level filters, leading to many conservative filters. Using

task-related web images can more effectively modify those filters for our task. The decrease of conservative filters is therefore a consequence of a more effective domain adaption process.

3.5 Experiments

Using insights from the experiments performed on UCF101 split1 in Section 4, we now perform experiments following the standard evaluation protocol [49] and report the average accuracy over the three provided splits.

We also perform experiments on ActivityNet. Following [14], we evaluate classification performance on both trimmed and untrimmed videos. Trimmed videos contain exactly one activity. Untrimmed videos contain one or more activities. We use the mAP (mean average precision) in evaluating performance. Results reported on ActivityNet are produced using the validation data, as the authors are reserving the test data for a potential future challenge.

3.5.1 Implementation

3.5.1.1 Experimental Setup for UCF101

Fine-tuning: We use the Caffe [48] software for fine-tuning CNNs. We use models VGG16, VGG19 [108], and M2048 [16] that are pre-trained on ImageNet by the corresponding authors. We only test M2048 on the first split for analysis, as it is shown to be significantly inferior to the other two architectures (Table 3.2). Due to hardware limitations, we use a small batch size: 20 for M2048 and 8 for VGG16 and VGG19. Accordingly, we use a smaller learning rate than those used in [16, 108]. For M2048, the initial learning rate 10^{-3} is changed to 10^{-4} after 40K iterations; training stops at 80K iterations. For both VGG16 and VGG19, the initial learning rate 10^{-4} is changed to 10^{-5} after 40K iterations, and is further lowered to 2×10^{-6} after 80K iterations. Training stops at 100K iterations.

Momentum and weight decay coefficients are always set to 0.9 and 5×10^{-4}. In each model, all layers are fine-tuned except the last fully connected layer which has to be changed to produce output of 101 dimensions with initial parameter values sampled from a zero-mean Gaussian distribution with $\sigma = 0.01$.

We resize video frames to 256×256, and random crops to 224×224 with random horizontal flipping for training. For web action images, since their aspect ratios vary significantly, we first resize the short dimension to 256 while keeping the aspect ratio, and subsequently crop six 256×256 patches along the longer dimension in equal spacing. Random cropping of 224×224 with random horizontal flipping is further applied to these image patches in training. Equal numbers of web images and video frames are sampled in each training batch.

Video Classification: A video is classified by fusing over the CNN outputs for the individual video frames. For a test video, we select 20 frames of equal temporal spacing. From each of the frames, 10 samples are generated following [66]: four corners and the center (each is 224×224) are first cropped from the 256×256 frame, making 5 samples; horizontal flipping of these samples makes another 5. Their classification scores are averaged to produce the frame's scores. We classify each frame to the class of the highest score, and the class of the video is then determined by voting of the frames' classes.

We also test SVM fusion, concatenating the CNN outputs for the 20 frames (averaged over the 10 cropped and flipped samples) from the second fully-connected layer (fc7), *i.e.* the 15th layer in VGG16 and 18th layer in VGG19. This produces a vector of 81,920 (4096×20) dimensions, which is then L2 normalized. One-vs-rest linear SVMs are then trained on these features for video classification. The SVM parameter $C = 1$ in all experiments.

Combining with Motion Features: The output of spatial CNNs can be combined with

motion features to achieve significantly better performance, as shown in [107]. We present an alternative by combining the output of the spatial CNNs with the conventional expert-designed features, namely the improved dense trajectories with Fisher encoding (IDT-FV) [124]. We follow the same settings in [124] to compute the IDT-FV for each video except that we do not use a space-time pyramid. The IDT-FV of each video is then combined with the concatenated fc7 outputs of 20 frames to form the final feature vector for a video. One-vs-rest linear SVMs are then trained on these features for video classification. The SVM parameter $C = 1$.

3.5.1.2 Experimental Setup for ActivityNet

We use the Caffe [48] software for fine-tuning CNNs. We use a VGG19 model [108] that is pre-trained on ImageNet by the authors. Due to hardware limitations, we use a small batch size of 8. Accordingly, we use a smaller learning rate than [108]. The initial learning rate 10^{-4} is changed to 10^{-5} after 80K iterations. Training stops at 160K iterations. Momentum and weight decay coefficients are set to 0.9 and 5×10^{-4}. All layers are fine-tuned except the last fully connected layer which has to be changed to produce output of 203 dimensions with initial parameter values sampled from a zero-mean Gaussian distribution with $\sigma = 0.01$.

Resizing and cropping of images and frames are performed in the same way as previously described for UCF101. Samples in each training batch are randomly selected from web action images and video frames with equal probability.

3.5.2 Results

3.5.2.1 Experimental Results for UCF101

Here we report the performance of our spatial CNNs averaged over three splits of UCF101 (Table 3.5), as well as the performance of our models when motion features are also used

Model	Accuracy (%)
slow fusion CNN [55]	65.4
spatial CNN [107]	73.0
VGG16, voting	77.9
VGG16 + Images, voting	82.5
VGG16 + Images, SVM fusion on fc7	**83.5**
VGG19, voting	77.8
VGG19 + Images, voting	83.3
VGG19 + Images, SVM fusion on fc7	83.4

Table 3.5: Mean accuracy of spatial CNNs (averaged over three splits) on UCF101.

(Table 3.6).

As seen in Table 3.5, all our spatial CNNs trained using both videos and images of BU101 improved $\sim$10% (absolute) in accuracy over the spatial CNN of [107], which is a 7-layer model. We believe this improvement is due to two main factors: using a deeper model and using web action images in training. Comparing the performance of the spatial CNN of [107] to the deeper models trained only on videos (rows 3 and 6 in Table 3.5), we find that the improvements solely due to differences of CNN architectures are 4.9% and 4.8% for VGG16 and VGG19 respectively. When web action images are used in addition to videos in training (rows 4 and 7 in Table 3.5), these improvements are doubled: 9.5% and 10.3% respectively.

Results reported in Table 3.5 show that, in the models we tested, the simple approach of using web action images in training contributes at least equally with introducing significant complexities to the CNN model, *i.e.*, adding at least 9 more layers. It is also interesting to note that, without using optical flow data, our spatial CNNs already approach performance attained using expert designed features that use optical flow, *i.e.* IDT-FV [124] in Table 3.6. Performance gains obtained by our approach are especially encouraging compared to deepening the model or incorporating motion features, as leveraging web images during training will not add any additional computational or memory burden during test time.

Model	Accuracy (%)
IDT-FV [124]	85.9
Two-stream CNN [107]	88.0
RCNN using LSTM [84]	88.6
MIFS [67]	89.1
C3D [116]	90.4
TDDs [128]	90.3
TDDs + IDT-FV [128]	91.5
VGG16 + Images + IDT-FV	91.1

Table 3.6: Mean accuracy (averaged over three splits) when combining spatial CNNs with motion features for UCF101.

The slow fusion CNN [55] is not a spatial CNN as it is trained on multiple video frames instead of single video frames. We list it here as it presents a different approach; collecting millions of web videos for training. However, despite the fact that 1M web videos are used as pre-training data, its performance is far lower than our models.

We further test the features learned by our spatial CNNs when combined with motion features, *i.e.* Fisher encoding on improved dense trajectories. Table 3.6 compares our results with state-of-the-art methods that also use motion features. Our method (VGG16 + Images + IDT-FV) outperforms all state-of-the-art temporal models (except TDDs + IDT-FV), improving by 2.5% over [84] that trains recurrent CNNs with long short-term memory cells; by 3.1% over [107], which combines two separate CNNs trained on video frames and optical flow respectively; by 5.2% over [124] that uses Fisher encoding on improved dense trajectories; by 2% over MIFS [67] that utilize multiple time skips to mimic multiple time-scales; by 0.7% over C3D [116] that use 3D convolution filters within a CNN model to learn space-time features; and by by 0.8% over TDDs [128] that use sum pooling of convolutional feature maps of trained two-stream ConvNets center-aligned on trajectories. When TDDs are used together with IDT-FV [128] we observe an improvement of 0.4% over our method. However, our method could, in turn, be used to further improve

Model	# Images	Untrimmed mAP (%)	Trimmed mAP (%)
fc8 [14]	none	25.3	38.1
DF [14]	none	28.9	43.7
Ours (video frames only)	none	52.3	47.7
Ours (unfiltered: all)	387K	53.8	49.5
Ours (unfiltered: rand select)	103K	53.3*	49.3*

Table 3.7: Although ActivityNet is large-scale, using unfiltered web images (BU203-unfiltered) still helps in both trimmed and untrimmed classification. * means average of three random sample sets.

this result further by training the spatial stream of [128] using both web images and videos.

3.5.2.2 Experimental Results for ActivityNet

We now test the performance of our spatial CNNs on ActivityNet for the task of action classification in trimmed and untrimmed videos with and without auxiliary web images (Table 3.7), *i.e.* images of BU203-unfiltered. We then further examine the use of web images as a substitute for many training videos (Table 3.8).

In Table 3.7 we observe that utilizing web images still helps $\sim$1.5% even with a very large scale dataset like ActivityNet. Using a random sample of approximately one quarter of the crawled web images gives nearly the same results, suggesting that performance gains diminish as the number of web action images greatly increase. This result is consistent with results on UCF101 (Figure 3.4).

In Table 3.8 we observe that comparable performance is achieved when half the training videos, are replaced by web images (rows 1 and 4 in Table 3.8). A similar pattern is observed when repeating the experiment at a smaller scale. This suggests that using a relatively small number of web images can help us reduce the effort of curating and storing millions of video frames for training.

Experiment	# Frames	# Images	mAP (%)
All vids	32.3M	none	47.7
1/2 vids	16.2M	none	40.9*
1/4 vids	8.1M	none	33.4*
1/2 vids + imgs	16.2M	387K	46.3*
1/4 vids + imgs	8.1M	387K	41.7*

Table 3.8: Comparable performance is achieved when half the training videos of ActivityNet are replaced by 393K images (row 4 *vs.* row 1). * means average of three random sample sets.

Figure 3.8: Sample video sequence with a ground truth label of *Happy* for the associated emotion. The frames of this video are sampled uniformly, the time dimension from left to right. Top: The original video frame, and Bottom: The pre-processed face.

3.6 Application to Facial Emotion Recognition

In this section, we show an application of web image augmentation to the task of video-based emotion recognition. The proposed approach takes the video stream of trimmed clips and produces the emotion label corresponding to this video sequence. This output is encoded as one out of seven classes: the six basic emotions (Anger, Disgust, Fear, Happiness, Sad, Surprise) and Neutral.

3.6.1 Datasets

Our target dataset for *video* recognition is The Acted Facial Expressions in the Wild (AFEW) 6.0 Dataset [25]. It consists of 1.4K trimmed video clips from movies annotated for facial expression. A sample video sequence from AFEW 6.0 is shown in Figure 3.8.

	Train	Valid	Test
Neutral	55180	1151	4396
Happy	26271	904	1801
Surprised	15421	422	725
Sad	11221	418	308
Angry	14063	305	843
Disgust	3372	19	87
Fear	5442	92	198

Table 3.9: Emotion category distribution of the image dataset

We augment the spatial classification of training video frames with a facial expression *images* dataset. This dataset was collected by crawling web images with various emotional keywords. The raw image set has over 4.5 million images. However, the majority of these images are either neutral or happy. 148K images were progressively selected for tagging, with the latter batches focusing more and more on rare emotions. Each image was annotated by 12-15 crowd workers into one of seven basic emotions (in addition to the 6 basic emotions that were mentioned earlier, we added contempt as the seventh emotion). The numbers of images per emotion category are summarized in Table 3.9.

3.6.2 Experimental Setup and Results

We use the video modality from the provided video-audio trimmed clips provided by the EmotiW'16 challenge. We do not use other modalities like audio, and we do not use any of the provided computed features. Our system consists of a face detection module, a pre-processing module, a deep feature extractor module, a feature encoding module, and finally an SVM classification module. Figure 3.9 summarizes the pipeline used to obtain our results.

We use the face detection approach of Chen *et al.* [18]. We then crop the frame to the largest face detection. We re-size the cropped face image to match the input size of our Convolutional Neural Networks (CNNs). We then convert image to grayscale, and perform

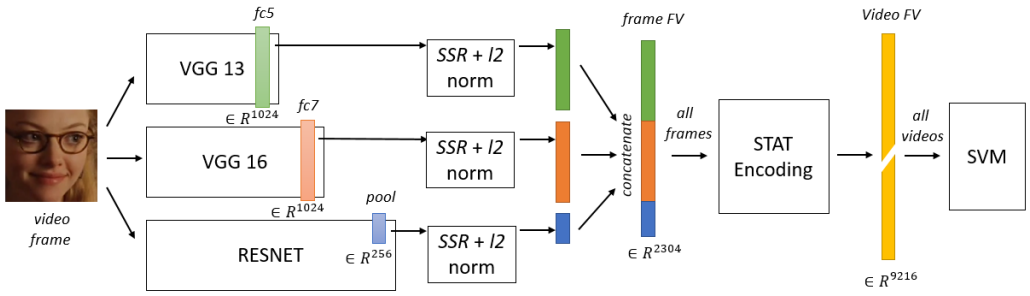


Figure 3.9: A depiction of the pipeline of our emotion recognition system. This depiction is specific to the combination of features that gave us best emotion recognition results: *fc5* of VGG13 + *fc7* of VGG16 + *pool* of ResNet. Each of there features is normalized using Signed Square Root(*SSR*) and *l2* normalization. The three normalized feature vectors are concatenated to create a single feature vector that describes this input frame. This is done for all frames of the video and inserted into the Statistical encoding module which produces a single feature vector representing the video. This feature is then used for SVM training or classification.

histogram equalization.

We train three networks: a modified VGG (13 layers) [10] based on [108], a second VGG (16 layers) [108], and ResNet (91 layers) [37]. Each of these networks is trained on the the combination of the image dataset and a set of sampled frames from the AFEW training set. We follow the Probabilistic Label Drawing training process recommended by Barsoum *et al.* [10] where a random emotion tag is drawn from the crowd-sourced label distribution of an image and used as the ground truth for that image in a certain epoch.

We then compute deep features using our learnt CNN models that were trained on images. We use the fully connected layer 5 (*fc5*) from the VGG13 network, the fully connected layer 7 (*fc7*) from the VGG16 network, and the global pooling layer (*pool*) from the ResNet network. For each video frame, we compute these three features: 1024-D *fc5* of VGG13, 1024-D *fc7* of VGG16, 256-D *pool* of ResNet. We normalize each of these features separately using Signed Square Root (*SSR*) and *l2* normalization.

Given the set of feature vectors representing the set of video frames, we encode these

Approach	Validation Acc (%)	Test Acc (%)
Challenge Baseline [24]	38.81	40.47
fc5 VGG13 + *fc7* VGG16 + *pool* ResNet	**59.42**	**56.66**

Table 3.10: Improvement over baseline by using a web image dataset to augment video frames in spatial convolutional neural networks for the task of emotion recognition for the EmotiW'16 Challenge.

features into a feature vector that represents the entire video sequence. This is done by computing and concatenating the mean, the variance, the minimum, and the maximum of feature dimensions over all video frames. This multiplies the dimensionality of the original feature vector by 4. We now normalize this encoded feature and use it for classification.

Encoded features computed as explained in section 3.3 are used to train a Support Vector Machine (SVM) to label each encoding with one of the 7 emotion classes. A One-*vs*-rest linear SVM is trained for classification using a grid search over the C paramter using 5-fold cross-validation. Best results were observed in the range $C \in [0.5, 2]$. Results reported here are using $C = 1$. We use sklearn's LinearSVC implementation that is based on liblinear. At test time, we compute the encoded features in the same way, and use the SVM class predictions.

Table 3.10 presents the margin of improvement over the Emotion Recognition in the Wild 2016 Challenge (EmotiW'16) baseline; This result ranked third place in EmotiW'16.

Improving model classification has benefits, but some misclassifications will persist. It is not possible from the current setup to reason why instances are correctly or incorrectly classified by our deep models. In the next chapter, we will explain how spatial grounding can help explain model decisions through visualization and demonstrate some applications of grounding models for visual data.

Chapter 4

Excitation Backprop

Saliency maps that quantize the importance of class-specific neurons for an input image are instrumental in this thesis. Popular approaches include Class Activation Maps (CAM) [156], Gradient-weighted Class Activation Mapping (Grad-CAM) [103], Randomized Input Sampling for Explanations (RISE) [93], Excitation Backprop (EB) [147]. EB is heavily used in this thesis since (a) it produces a valid probability distribution for each network layer, (b) it has a contrastive formulation that results in discriminative evidence for a specific class, and (c) we focus on improving white-box models of known architecture and weights.

In this chapter, a brief background on Excitation Backprop (EB) [148] is given. EB devises a backpropagation formulation able to reconstruct the evidence used by a deep model to make decisions in the form of probability distributions that can be visualized as saliency maps. EB passes top-down signals, a prior distribution over the output units, through excitatory connections. Recursively propagating the top-down signal and preserving the sum of backpropagated probabilities layer by layer, it computes task-specific saliency maps from any intermediate layer in a single backward pass.

In a standard CNN, the forward activation of neuron a_j in a CNN is computed by $\widehat{a}_j = \phi(\sum_i w_{ij}\widehat{a}_i + b_i)$, where $\widehat{a}_i$ is the activation coming from a lower layer, ϕ is a nonlinear activation function, w_{ij} is the weight from neuron i to neuron j, and b_i is the added bias at layer i. The EB framework makes two key assumptions about the activation

$\widehat{a}_j$ which are satisfied in the majority of modern CNNs due to wide usage of the *ReLU* non-linearity:

A1. $\widehat{a}_j$ is non-negative

A2. $\widehat{a}_j$ is a response that is positively correlated with its confidence of the detection of specific visual features.

EB realized a probabilistic Winner-Take-All (WTA) formulation to efficiently compute the probability of each neuron recursively using conditional probabilities $P(a_i|a_j)$ in a top-down order starting from a probability distribution over the output units, as follows:

$$P(a_i) = \sum_{a_j \in \mathcal{P}_i} P(a_i|a_j)P(a_j) \tag{4.1}$$

where $\mathcal{P}_i$ is the parent node set of a_i. EB passes top-down signals through excitatory connections having non-negative activations, excluding from the competition inhibitory ones. Assuming C_j the child node set of a_j, for each $a_i \in C_j$, the conditional winning probability $P(a_i|a_j)$ is defined as

$$P(a_i|a_j) = \begin{cases} Z_j \widehat{a}_i w_{ij}, & \text{if } w_{ij} \geq 0, \\ 0, & \text{otherwise} \end{cases} \tag{4.2}$$

where Z_j is a normalization factor such that $\sum_{a_i \in \mathcal{C}_j} P(a_i|a_j) = 1$. Recursively propagating the top-down signal and preserving the sum of backpropagated probabilities, it is possible to highlight the salient neurons in each layer using Equation 4.1, *i.e.* neurons that mostly contribute to a specific task. This is depicted in Figure 4.1. We will refer to the distribution of $P(a_i)$ as $p_{EB}(a_i)$.

To improve the discriminativeness of the saliency maps, [148] introduced *contrastive*

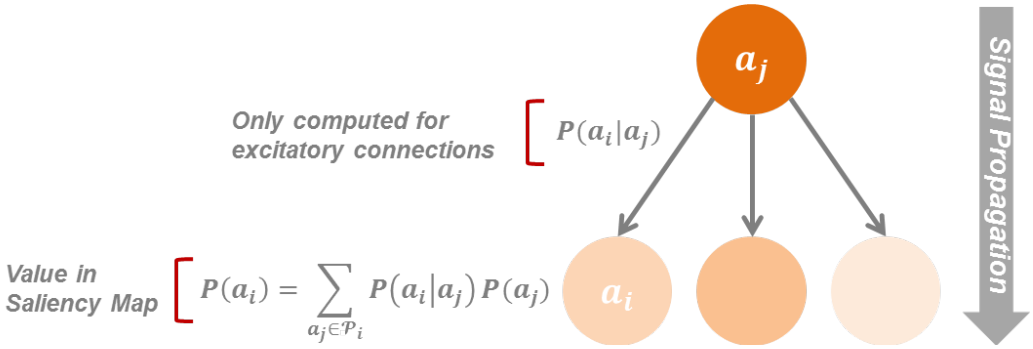


Figure 4.1: In Excitation Backprop, excitation probabilities are propagated in a single backward pass in the CNN. A top-down signal is a probability distribution over the output units. The probabilities are backpropagated from every parent node to its children through its excitatory connections. The figure illustrates the contributions of a single parent neuron to the excitation probabilities computed at the next layer. Each $P(a_i)$ in the saliency map is computed over the complete parent set $\mathcal{P}_i$. Shading of nodes in the figure conveys $P(a_i)$ (darker shade = greater $P(a_i)$).

EB (cEB) which cancels out common winner neurons and amplifies the class discriminative neurons. To do this, given an output unit $o_i \in O$, a dual unit $\overline{o}_i \in \overline{O}$ is virtually generated, whose input weights are the negation of those of o_i. By subtracting the saliency map for $\overline{o}_i$ from the one for o_i the result better highlights cues in the image that are unique to the desired class. Figure 4.2 demonstrates the computation of such discriminative maps.

4.1 Example Applications

In this section, we demonstrate how the identification of evidence within a visual input using top-down neural attention formulations can be a powerful tool for model interpretation, computer-aided annotation, and domain analysis.

4.1.1 Model Interpretation and Data Annotation

In this section we discuss applications of the EB neural attention method in model interpretation and data annotation. In tasks like medical image analysis or fine-grained image

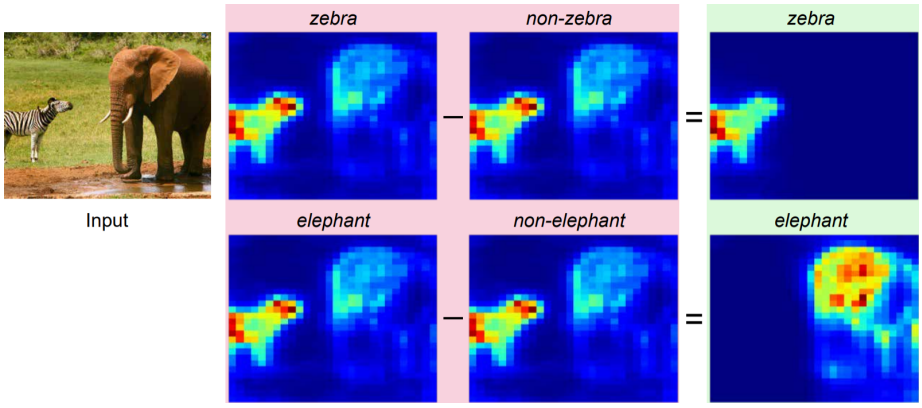


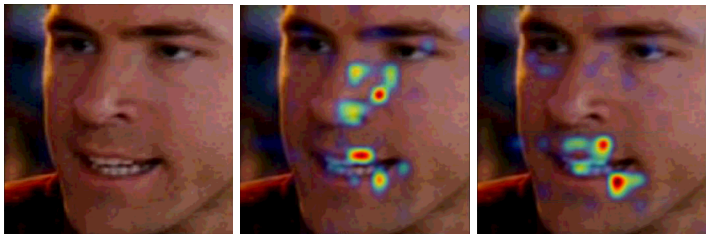
Figure 4.2: Excitation Backprop(EB) *vs. contrastive* Excitation Backprop (*c*EB). The input image is resized to 224×224, and we use VGG16 pretrained on ImageNet to generate the EB maps for *Zebra* and *Elephant*, as well as *non-Zebra* and *non-Elephant*. The *c*EB maps (green) are computed by subtracting the *non-Zebra* (*non-Elephant*) EB map from the *Zebra* (*Elephant*) EB map (pink), and then thresholding the values at 0. All attention maps are rescaled for visualization. The EB maps shown above look nearly identical. Their subtle differences are captured by the *c*EB maps, where the common winner neurons for different concepts are cancelled out.

classification, grounding a neural network model's prediction can not only help users better understand the mechanism and capability of the model, but also provide guidance for data annotation. We provide two examples of such applications.

Facial Expression Analysis. We demonstrate sample analysis of correct/incorrect predictions of a VGG-S model trained to classify a face image into one of the six basic facial emotions [69] (Angry, Happy, Sad, Surprise, Disgust, Fear), and Neutral. This model is trained on the training set of the Static Facial Expressions in the Wild (SFEW) dataset [25].

Using *c*EB maps, we can visualize local evidence used by our model for a target emotion category. Sample analysis of correct and incorrect classifications of the model using validation images from the SFEW dataset are demonstrated in Figure 4.3. In Figure 4.3 (a), the *c*EB for the correctly classified category "Happy" shows that the model uses the evidence around the mouth. In Figure 4.3 (b), the *c*EB maps for "Angry" (ground truth) and

(a) correctly classified example



(b) Incorrectly classified example

Figure 4.3: This figure shows (a) a correctly classified example (“Happy"), together with the evidence the model uses for the classification, an (b) an incorrectly classified example (left) with a ground truth label “Angry". This image is mis-classified by the trained model as “Happy". We demonstrate using *c*EB why the network thinks this is “Happy" (right). We also show the *c*EB map of “Angry" (middle).

“Happy" (model’s prediction) give different focused regions. It indicates that for “Angry", the model is looking for evidence around the nose and eyes in addition to the mouth. This type of visualization can also be useful for human annotator to annotate facial action units (*e.g.* Chin Raiser, Nose Wrinkler, Outer Brow Raiser, Jaw Drop) under the Facial Action Coding System[1]. Annotating such facial action units requires training. The top-down attention maps can help non-professional annotators localize action units that are relevant to a high-level emotion.

Medical Image Analysis. Medical image analysis tasks are usually quite demanding. Machine learning methods can help speedup these tasks, but in many scenarios human experts need to examine the predicted results. Thus, it is of special importance that machine

[1]The Facial Action Coding System (FACS) is a taxonomy for encoding facial muscle movements into Action Units (AUs). Combinations of coded action units are used to make higher-level decisions, such as a facial emotion: happy, sad, angry, etc.

learning models can point human experts to the relevant regions that may support or reject the predicted results.

Huang *et al.* [42] use a neural network model to label fetal heart orientation on ultrasound images. To verify their model learns the task-specific features, they use Excitation Backprop to produce attention maps for their model.

Jamaludin *et al.* [46] propose a neural network model to automatically produce radiological gradings of spinal lumbar MRIs. They demonstrate that the *c*EB maps of their model generated by our method can clearly localize pathological regions in the disc volumes, even although no segmentation annotation is used during their model training.

4.1.2 Domain Analysis

We now explore how *c*EB can be used to highlight discriminative evidence found in each domain in a domain transfer setting.

Highlighting Domain Evidence. We use *c*EB to visualize regions responsible for domain transfer. Having a classifier trained to differentiate between domains, we can then visualize why the model thinks an image belongs to a specific domain and not the other. We use the VisDA (Visual Domain Adaptation) dataset [91], which is constructed from a graphics source domain and a real images target domain. We train a VGG16 network to differentiate between the graphics and real images domains of the VisDA dataset. As *c*EB can be used to visualize classes that are not necessarily ground-truth, we visualize the evidence for each domain in images of the source domain and images of the target domain in Figure 4.4 and Figure 4.5, respectively. Figure 4.4 shows that the model uses the white background as evidence for the graphics domain, and the object as evidence for the real images domain. Figure 4.5 shows that the model uses the object and strong shadows as evidence for the graphics domain, and the busy background as evidence for the real images

domain. This capability of interpreting models and visually analyzing differences between domains suggests the possibility of building models that bridge exactly that highlighted domain gap.

Highlighting the Evidence Shift. We now use *c*EB to highlight the shift of focus in the input images when different training strategies are used. The first training strategy is vanilla training with no domain adaptation, *i.e.* training on graphics images only and testing on real images only. The second training strategy employs domain adaptation. We highlight the shift of evidence in a test image that was misclassified by a model that does not perform domain adaptation, and was then correctly classified by a domain adaptation model. We train an Alex-Net on the VisDA classification task without Domain Adaptation. We then repeat training with the domain adaptation approach of Long *et al.* [75]. This approach aligns distributions of the source and target domains based on based on a joint maximum mean discrepancy. We then visualize the model evidence before and after domain adaptation. Examples are presented in Figure 4.6 demonstrating the shift toward more discriminative evidence. For example, before domain adaptation a metallic surface of the airplane was the evidence the network used to incorrectly classify the airplane as a motorcycle. However, after domain adaptation the image is correctly classified as an airplane and the evidence has shifted to the wings of the airplane.

4.2 Discussion

In this chapter we demonstrated how grounding is a useful tool in visualizing cues of the evidence used by models to make correct predictions, the evidence used by models to make incorrect predictions, and how different training strategies make models reason based on different evidence. While visualizations are great for qualitative analysis of individual instances, it is challenging to quantitatively measure the benefit of spatial grounding for

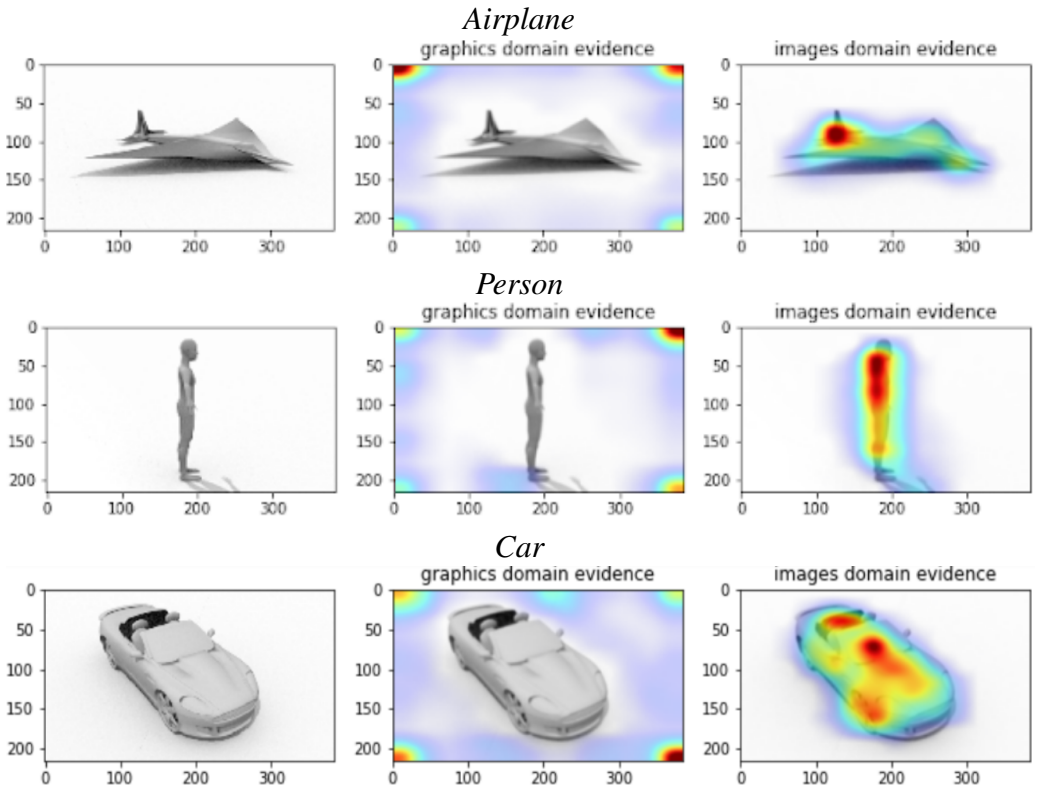


Figure 4.4: In the leftmost column we show images of an airplane, a person, and a car from the graphics source domain of the VisDA dataset. In the middle column we show the evidence the domain discrimination network would use to classify each of the images as graphics images. In the rightmost column we show the evidence the domain discrimination network would use to classify each of the images as real images.

model improvement in this way. In the next chapter, we show how spatial grounding can be used to develop a novel training regularizer that improves the generalization ability of deep models, increases the utilization of network neurons, and increases network resilience to compression.

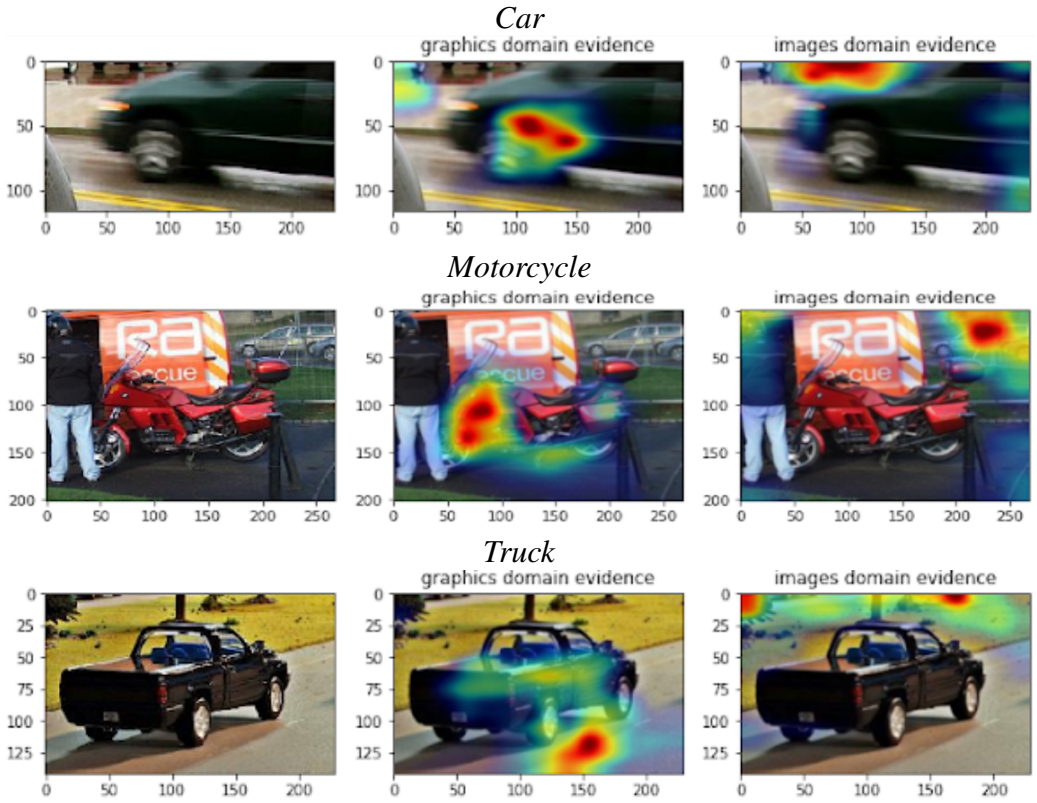


Figure 4.5: In the leftmost column we show images of a car, a motorcycle, and a truck from the real images target domain of the VisDA dataset. In the middle column we show the evidence the domain discrimination network would use to classify each of the images as graphics images. In the rightmost column we show the evidence the domain discrimination network would use to classify each of the images as real images.

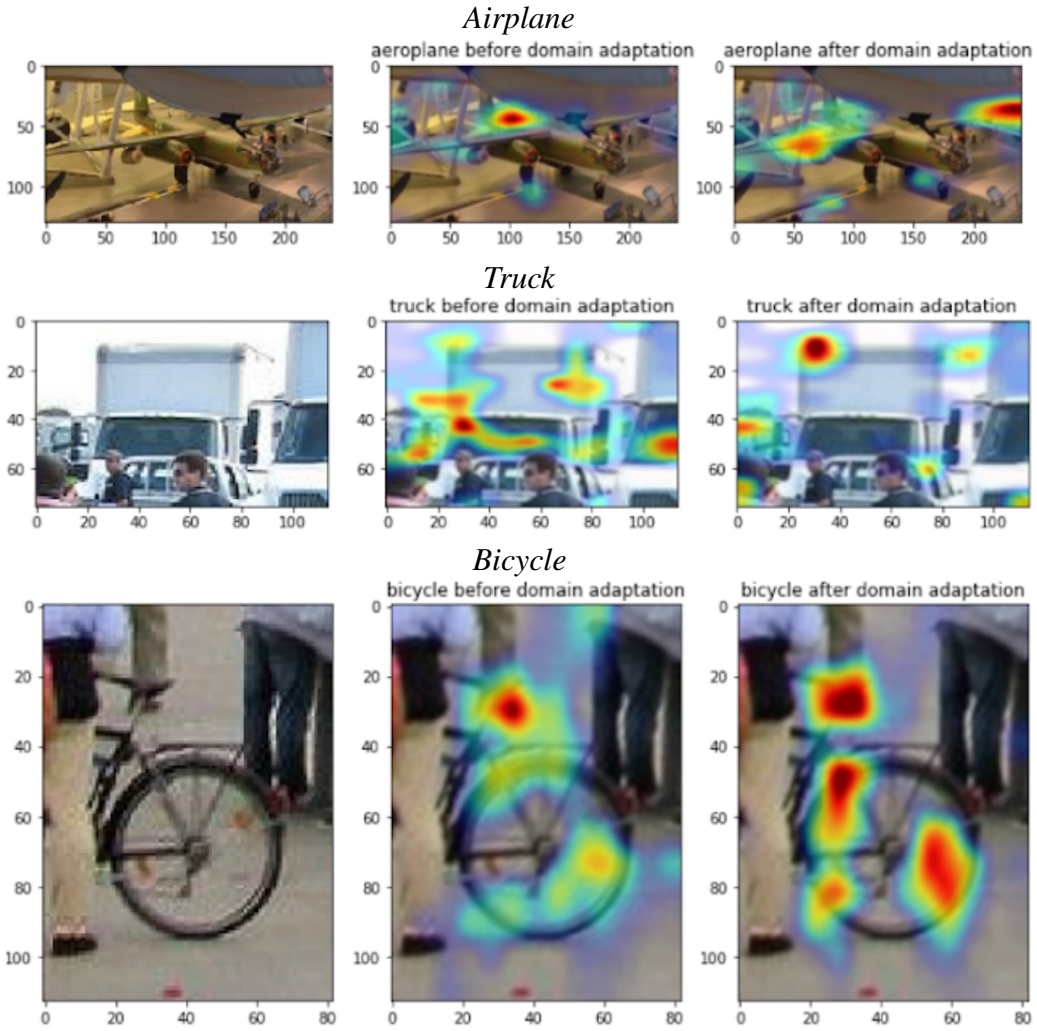


Figure 4.6: On the left we show images from the target domain that were misclassified before domain adaptation then were correctly classified after domain adaptation. We visualize how the model evidence shifts from an incorrect prediction (middle) to a correct one (right). It is clear how the evidence, after domain adaptation, is more focused on discriminative evidence of the ground-truth object class.

Chapter 5

Excitation Dropout: Using Spatial Saliency to Encourage Plasticity in Deep Neural Networks

Dropout [40, 112] is a classical regularization technique that is used in many state-of-the-art deep neural networks, typically applied to fully-connected layers. Standard Dropout selects a fraction of neurons to randomly drop out by zeroing their forward signal. In this chapter, we propose a scheme for biasing this selection. Our scheme utilizes the contribution of neurons to the prediction made by the network at a certain training iteration stage.

Dropout can be interpreted as model averaging technique that avoids overfitting on training data, allowing for better generalization on unseen test data. A recent variant of dropout that targets improved generalization ability is Curriculum Dropout [83]. It targets adjusting the dropout rate by exponentially increasing the unit suppression rate during training, answering the question *How many neurons to drop out over time?* Like Standard Dropout [40, 112], Curriculum Dropout selects the neurons to be dropped randomly. In this chapter, however, we target at determining how the dropped neurons are selected, answering the question *Which neurons to drop out?*

Our approach is inspired by brain plasticity [38, 109, 80, 79]. We deliberately, and temporarily, paralyze/injure neurons to enforce learning alternative paths in a deep network. At training time, neurons that are more relevant to the current prediction are given a higher dropout probability. The relevance of a neuron for making a certain prediction is quantified

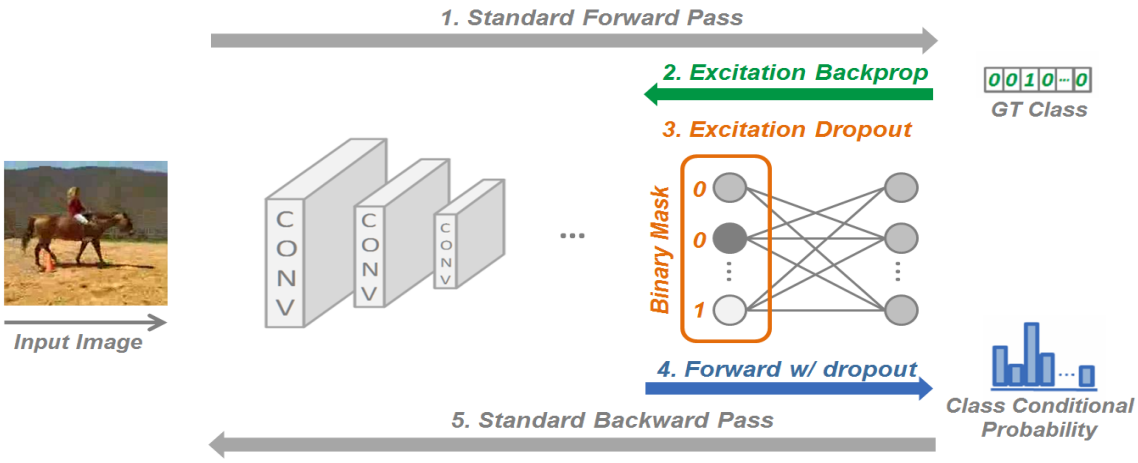


Figure 5.1: Training pipeline of Excitation Dropout. *Step 1:* A minibatch goes through the standard forward pass. *Step 2:* Backward EB is performed until the specified dropout layer; this gives a neuron saliency map at the dropout layer in the form of a probability distribution. *Step 3:* The probability distribution is used to generate a binary mask for each image of the batch based on a Bernoulli distribution determining whether each neuron will be dropped out or not. *Step 4:* A forward pass is performed from the specified dropout layer to the end of the network, zeroing the activations of the dropped out neurons. *Step 5:* The standard backward pass is performed to update model weights.

using Excitation Backprop, a top-down saliency approach proposed by Zhang *et al.* [148]. Excitation Backprop conveniently yields a probability distribution at each layer that reflects neuron saliency, or neuron contribution to the prediction being made. This is utilized in the pipeline of our approach, named Excitation Dropout, which is summarized in Figure 5.1.

In particular, we study how this approach improves generalization through utilizing more network's neurons for image classification. We report an increased recognition rate for both CNN models that are fine-tuned and trained from scratch. This improvement is validated on four image/video recognition datasets, and ranges from 1.1% - 6.3% over state-of-the-art Curriculum Dropout.

Next, we examine the effect of our approach on network utilization. Mittal *et al.* [80] and Ma *et al.* [76] introduce metrics that measure network utilization. We show a consistent increased network utilization using Excitation Dropout on four image/video recognition

datasets. For example, averaged over all four benchmarks, we get 76.55% reduction in conservative filters, filters whose parameters do not change significantly during training, as compared to Standard Dropout.

Finally, we study network resilience to neuron dropping at test time. We observe that training with Excitation Dropout leads to models that are a lot more robust when layers are shrunk/compressed by removing units. We demonstrate this when dropping the most relevant neurons, the least relevant neurons, and with a random dropping selection. This can be quite desirable for compressing/distilling [39] a model, *e.g.* for deployment on mobile devices.

In summary, by encouraging plasticity-like behavior, our contributions are threefold:

1. Better generalization on test data.
2. Higher utilization of network neurons.
3. Resilience to network compression.

5.1 Method

In the standard formulation of dropout [40, 112], the suppression of a neuron in a given layer is modeled by a Bernoulli random variable $0 < p \leq 1$ where p is defined as the probability of retaining a neuron. Given a specific layer where dropout is applied, during the training phase, each neuron is turned off with a probability $1 - p$.

We argue for a different approach that is *guided* in the way it selects neurons to be dropped. In a training iteration, certain paths have high excitation contributing to the resulting classification, while other regions of the network have low response. We encourage learning alternative paths (plasticity) through the temporary damaging of the currently highly excited path. We re-define the probability of retaining a neuron as a

function of its contribution in the currently highly excited path

$$p = 1 - \frac{(1-P) * (N-1) * p_{EB}}{((1-P) * N - 1) * p_{EB} + P} \tag{5.1}$$

where N is the number of neurons in layer l, p_{EB} is the probability backpropagated through the EB formulation (Equation (4.1)) in layer l, and P is the *base* probability of retaining a neuron when all neurons are equally contributing to the prediction. The retaining probability defined in Equation (5.1) drops neurons which contribute the most to the recognition of a specific class, with higher probability. Dropping out highly relevant neurons, we retain less relevant ones and thus encourage them to awaken. We also study how this compares to dropping the least relevant neurons (Adaptive Dropout by [4]) in Section 5.2.2.

Figure 5.2 shows p as a function of p_{EB}. To gain some intuition for Equation (5.1), we can look more closely at the graph: 1) If neuron a_i has $p_{EB}(a_i) = 1$: This results in a retaining probability of $p = 0$. We do not want to keep a neuron which has a high contribution to the correct label. 2) If neuron a_i has $p_{EB}(a_i) = 0$: This results in a retaining probability of $p = 1$. We want to keep a neuron which has not contributed to the correct classification of an image. 3) If neuron a_i has $p_{EB}(a_i) = 1/N$, *i.e.* p_{EB} is a uniform probability distribution: This results in a retaining probability $p = P$. We want to keep a neuron with *base* probability P since all neurons contribute equally.

Equation (5.1) provides a dropout probability for each neuron, which is then used as the parameter of a Bernoulli distribution giving a binary dropout mask. During training, each image in a batch leads to different excitatory connections in the network and therefore has a different p_{EB} distribution, consequently leading to a different dropout mask. Figure 5.1 presents the pipeline of Excitation Dropout at training time.

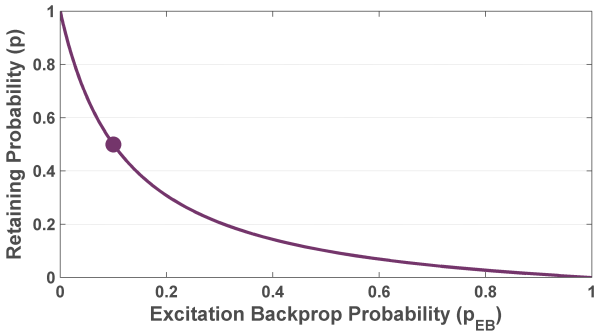


Figure 5.2: The retaining probability, p, as a function of the Excitation Backprop probability p_{EB}. This plot was created using $N = 10$ and a base retaining probability $P = 0.5$. In this case, when the saliency of neurons is uniform, *i.e.* $p_{EB} = 0.1$, then $p = P$ as marked in the figure.

5.2 Experiments

In this section, we present how Excitation Dropout (ED) improves the generalization ability on four image/video recognition datasets on different architectures. We then present an analysis of how ED affects the utilization of network neurons on the same datasets. Finally, we examine the resilience of a model trained using Excitation Dropout to network compression.

5.2.1 Datasets and Architectures

We present results on four image/video recognition datasets. **Cifar10** and **Cifar100** [65] are image recognition datasets, each consisting of 60000 32×32 tiny RGB natural images. Cifar10 images are distributed over 10 classes with 6000 images per class, and Cifar100 images are distributed over 100 classes with 600 images per class. Training and test splits contain $50K$ and $10K$ images, respectively. We feed the network with the original image dimensions. **Caltech256** [34] is an image recognition dataset consisting 31000 RGB images divided in 256 classes. We consider 50 train images and 20 testing images for each class. Images were reshaped to 128×128 pixel to feed the network. **UCF101** [110] is a video

action recognition dataset based on 13320 actions belonging to 101 action classes. For this dataset we consider a frame-based action recognition task. The images are resized to 224×224 and 227×227 to fit the input layers of the VGG and AlexNet architectures, respectively.

We present results on four architectures. Relatively shallow architectures are trained from scratch, and deeper popular architectures are fine-tuned after being pre-trained on ImageNet [20]. **Models trained from scratch:** We train the CNN-2 architecture used in [83], the state-of-the-art dropout variant, for comparison purposes. This architecture consists of three convolutional and two fully-connected layers. We train this network from scratch for $100K$ iterations on the datasets: Cifar-10, Cifar-100 and Caltech-256. We use mini-batches of 100 images and fix the learning rate to be 10^{-3}, decreasing to 10^{-4} after $25K$ iterations. **Fine-tuned models:** We fine-tune the commonly used architectures: AlexNet [66], VGG16 and VGG19 [108] pre-trained on ImageNet. We fine-tune the models for a frame by frame action recognition task on UCF101. The learning rate is fixed to 10^{-3} for all the processes. We fine-tune AlexNet for $5K$ while VGG16 and VGG19 for $30K$ iterations. We use a batch size of 128 and 50 images for AlexNet and VGG16/19, respectively.

5.2.2 Setup and Results: Generalization

In this section we compare the performance of ED to that of No Dropout, Standard Dropout, and state-of-the-art Curriculum Dropout [83]. We train a CNN-2 model from scratch on the datasets: Cifar10, Cifar100, Caltech256. Figure 5.3 depicts the test accuracies over training iterations for the three datasets averaged over five trained models. After convergence, ED demonstrates a significant improvement in performance compared to other methods. We hypothesize that ED takes longer to converge due to the additional loop (Steps 2-4 in

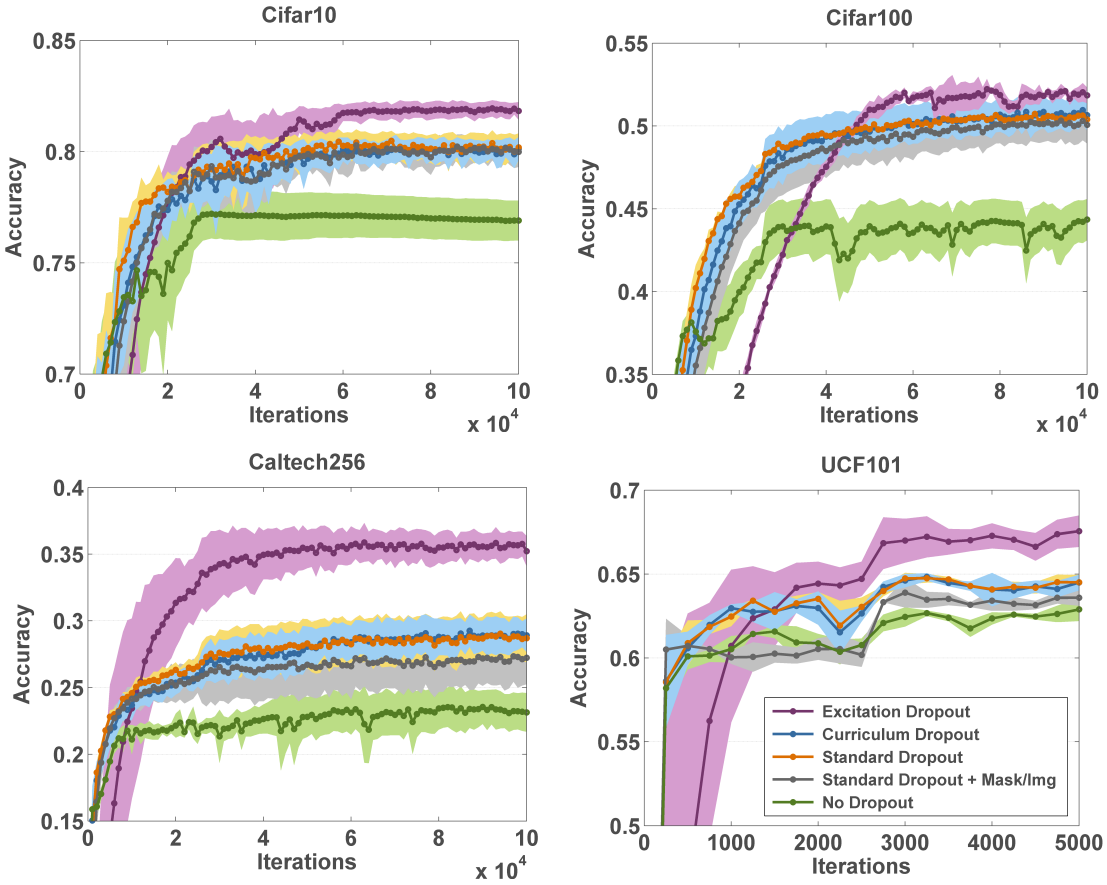


Figure 5.3: We compare the test accuracy of different dropout training strategies on four image/video recognition datasets: Cifar10, Cifar100, Caltech256, UCF101. Results presented here are averaged over five trained models and the standard deviation is depicted around the mean curve using a lighter shade. Excitation Dropout performs best after convergence compared to the other strategies.

Figure 5.1) introduced in the learning process, and due to the learning of the alternative paths. We note that ED, during training, uses a different binary mask for each image in a minibatch, while in Standard Dropout, one random mask is employed per minibatch. To prove that the actual boost in accuracy with ED is not provided by the choice of specific masks, we add a comparison with Standard Dropout having a different random mask for each image. We refer to this accuracy as 'Standard Dropout + Mask/Img' in the plots. As expected, the latter approach is comparable to Standard Dropout in performance.

Dataset	Adaptive Dropout	Excitation Dropout
Cifar10	76.82%	**81.94%**
Cifar100	44.55%	**52.04%**
Caltech256	23.32%	**35.77%**
UCF101	71.76%	**73.23%**

Table 5.1: Comparison between Adaptive Dropout and Excitation Dropout. The numbers reported in this table are the average test set accuracy over five trained models for each dataset.

Next, we compare methods that drop least *vs.* most relevant neurons. Popular dropout methods (*e.g.* Adaptive Dropout [4]) drop *useless* neurons with low activations during training. In this chapter we motivate and demonstrate that dropping neurons based on their Excitation Backprop (EB) probability has added benefits. Please note that we are not simply considering neuron activation. We demonstrate the performance of ED compared to the variant Adaptive Dropout [4] in Table 5.1. In essence, Adaptive and Excitation Dropout are opposites by dropping the least and most important neurons, respectively.

Next, we evaluate the effectiveness of ED on popular network architectures that employ dropout layers: AlexNet, VGG16, VGG19. This is done by fine-tuning on the video recognition test dataset UCF101. Figure 5.3 shows superior ED performance on AlexNet fine-tuned on UCF101. Table 5.2 presents more comparative results on other deep architectures by reporting the accuracy after convergence. Again, ED demonstrates higher generalizability on the test data for all considered architectures.

For fair comparison, we set $p = 0.5$ for Standard Dropout and $P = 0.5$ for the base retaining probability of ED in all experiments. We perform dropout in the first fully-connected layer of the considered networks (*fc1* for CNN-2 and *fc6* for AlexNet and VGGs) for Standard, Curriculum, and Excitation Dropout. For Curriculum Dropout we fix the parameter γ to $5 * 10^{-4}$ as in [83].

Architecture	No Dropout (%)	Standard Dropout (%)	Curriculum Dropout (%)	Excitation Dropout (%)
VGG16	69.37	71.93 (+2.56%)	72.14 (+2.77%)	**73.23 (+3.86%)**
VGG19	71.32	72.52 (+1.29%)	73.18 (+1.86%)	**74.34 (+3.02%)**
AlexNet	62.89	64.50 (+1.61%)	64.55 (+1.66%)	**67.56 (+4.67%)**

Table 5.2: Test accuracy comparison between No, Standard, Curriculum and Excitation Dropout in the $fc6$ layer of three architectures: AlexNet [66], VGG16 and VGG19 [108], fine-tuned for the action recognition task on UCF101 [110]. The numbers reported are the final test accuracies together with the improvements (in parenthesis) with respect to No Dropout, averaged over five trained models.

5.2.3 Setup and Results: Utilization of Network Neurons

In this section we examine how ED expands the network's utilization of neurons through the ability of re-wiring or having multiple paths for a certain task.

Mittal *et al.* [80] introduced scoring functions to rank the filters in specific network layers including the *average percentage of zero activations*, a metric to count how many neurons have zero activations, and the *entropy of activations*, a metric to measure how much information is contained in the neurons of a layer. We analogously compute the *entropy of* p_{EB} which is higher when the probability distribution is spread out over more neurons in a layer. We also compute the *peak* p_{EB} which is expected to be lower on a more spread distribution. Moreover, Ma *et al.* [76] introduced *conservative filters*: filters whose parameters do not change significantly during training. Conservative filters reduce the effective number of parameters in a CNN and may limit the CNN's modeling capacity for the target task. A conservative filter is a filter k in layer n whose weights have changed by $\Delta_n^k = \|\hat{w}_n^k - w_n^k\|$, where Δ_n^k is less than a threshold Δ.

We evaluate the presented metrics for Excitation Dropout and compare against Standard and Curriculum Dropout in Table 5.3. This is done on the same datasets and architectures considered in Sec. 5.2.2. All metrics are computed for the first fully-connected layer of the CNN-2 and VGG16 nets consisting of 2048 and 4096 neurons, respectively. We compute

Dataset	Metric	Standard Dropout	Curriculum Dropout	Excitation Dropout
Cifar10	# Neurons ON	1194 ($\pm$153)	1169 ($\pm$61)	**1325** ($\pm$61)
	Peak p_{EB}	0.011 ($\pm$0.004)	0.009 ($\pm$0.001)	**0.003** ($\pm$0.0002)
	Entropy of Activations	3.55 ($\pm$0.72)	3.50 ($\pm$0.12)	**4.29** ($\pm$0.28)
	Entropy of p_{EB}	3.28 ($\pm$0.56)	3.32 ($\pm$0.13)	**4.26** ($\pm$0.26)
	Cons. Filters$_{\Delta=0.25}$	1204 ($\pm$37)	959 ($\pm$34)	**124** ($\pm$22)
Cifar100	# Neurons ON	453 ($\pm$183)	460 ($\pm$75)	**943** ($\pm$131)
	Peak p_{EB}	0.011 ($\pm$0.0004)	0.012 ($\pm$0.0004)	**0.005** ($\pm$0.0005)
	Entropy of Activations	1.67 ($\pm$0.31)	1.70 ($\pm$0.29)	**3.21** ($\pm$0.44)
	Entropy of p_{EB}	1.64 ($\pm$0.27)	1.67 ($\pm$0.26)	**3.17** ($\pm$0.41)
	Cons. Filters$_{\Delta=0.30}$	2048 ($\pm$51)	2038 ($\pm$44)	**14** ($\pm$13)
Caltech256	# Neurons ON	412 ($\pm$126)	471 ($\pm$146)	**702** ($\pm$171)
	Peak p_{EB}	0.014 ($\pm$0.0007)	0.013 ($\pm$0.0006)	**0.007** ($\pm$0.0003)
	Entropy of Activations	1.63 ($\pm$0.32)	1.84 ($\pm$0.35)	**2.63** ($\pm$0.23)
	Entropy of p_{EB}	1.58 ($\pm$0.29)	1.77 ($\pm$0.31)	**2.59** ($\pm$0.22)
	Cons. Filters$_{\Delta=1.25}$	2048 ($\pm$46)	2048 ($\pm$49)	**1671** ($\pm$31)
UCF101	# Neurons ON	1120 ($\pm$25)	1143 ($\pm$22)	**1404** ($\pm$37)
	Peak p_{EB}	0.007 ($\pm$0.0002)	0.007 ($\pm$0.0002)	**0.004** ($\pm$0.0002)
	Entropy of Activations	2.04 ($\pm$0.23)	2.08 ($\pm$0.21)	**2.51** ($\pm$0.18)
	Entropy of p_{EB}	1.92 ($\pm$0.22)	1.95 ($\pm$0.20)	**2.42** ($\pm$0.18)
	Cons. Filters$_{\Delta=0.15}$	3599 ($\pm$66)	3859 ($\pm$53)	**44** ($\pm$36)

Table 5.3: Different metrics to reflect the usage of network capacity in the first fully-connected layer of the CNN-2 architecture consisting of 2048 neurons and the VGG16 consisting of 4096 neurons. Results presented here are averaged over five trained models for each of the datasets: Cifar10, Cifar100, Caltech256 and UCF101 (σ in brackets). Excitation Dropout consistently produces more neurons with non-zero activations, has a more spread saliency map leading to a lower saliency peak, has a higher entropy of both activations and saliency, and has a lower number of conservative (Cons.) filters; all reflecting an improved utilization of the network neurons using Excitation Dropout.

each metric over the test set of each dataset. Excitation Dropout consistently outperforms Standard and Curriculum Dropout in all the metrics over all datasets. ED shows a higher number of active neurons, a higher entropy over activations, a probability distribution p_{EB} that is more spread (higher entropy over p_{EB}) among the neurons of the layer, leading to a lower peak probability of p_{EB}. We also observe a significantly smaller number of conservative filters when using ED. Fewer filters remain unchanged, *i.e.* do not sufficiently

learn anything far from the random initialization. These results show that the models trained with ED were trained to be more informative, *i.e.* the contribution for the final classification task is provided by a higher number of neurons in the network, reflecting the alternative learnt paths.

5.2.4 Setup and Results: Resilience to Compression

In this section, we simulate 'Brain Damage' by dropping out neurons at test time. Figure 5.4 demonstrates a network re-wiring itself in order to capture the evidence of the class *HorseRiding* in a video frame of the UCF101 dataset. Given a VGG16 model fine-tuned with Excitation, Curriculum, Standard, and No Dropout at the $fc6$ layer, we show the excitation saliency map obtained at the conv5-1 layer as we drop out a fixed number of the most relevant neurons from the same layer dropout is performed upon during training. A neuron is considered to be more relevant if it has a higher p_{EB}. In the first column of Figure 5.4, the original saliency maps for the different models are shown. As already highlighted in Table 5.3, the original saliency map obtained from the model trained with Excitation Dropout is more spread as compared to that of the other schemes, which present more pronounced red peaks. In the following columns (from 2nd to 6th) of Figure 5.4, we present the saliency maps the model is able to restore when the $100, 200, 300, 400, 500$ most relevant neurons are dropped out. Despite the increasing number of relevant neurons being dropped out, ED is capable of restoring more of the saliency map contributing to *HorseRiding*. This means that the network with ED was trained to find alternative paths which belong to the same *HorseRiding*-relevant cues of the image. Despite the fact that we are considering the *worst-case* scenario, where we are switching off the *most* relevant neurons at test time, ED shows most robustness.

While Figure 5.4 visualizes one example qualitatively, Figure 5.5 presents a complete

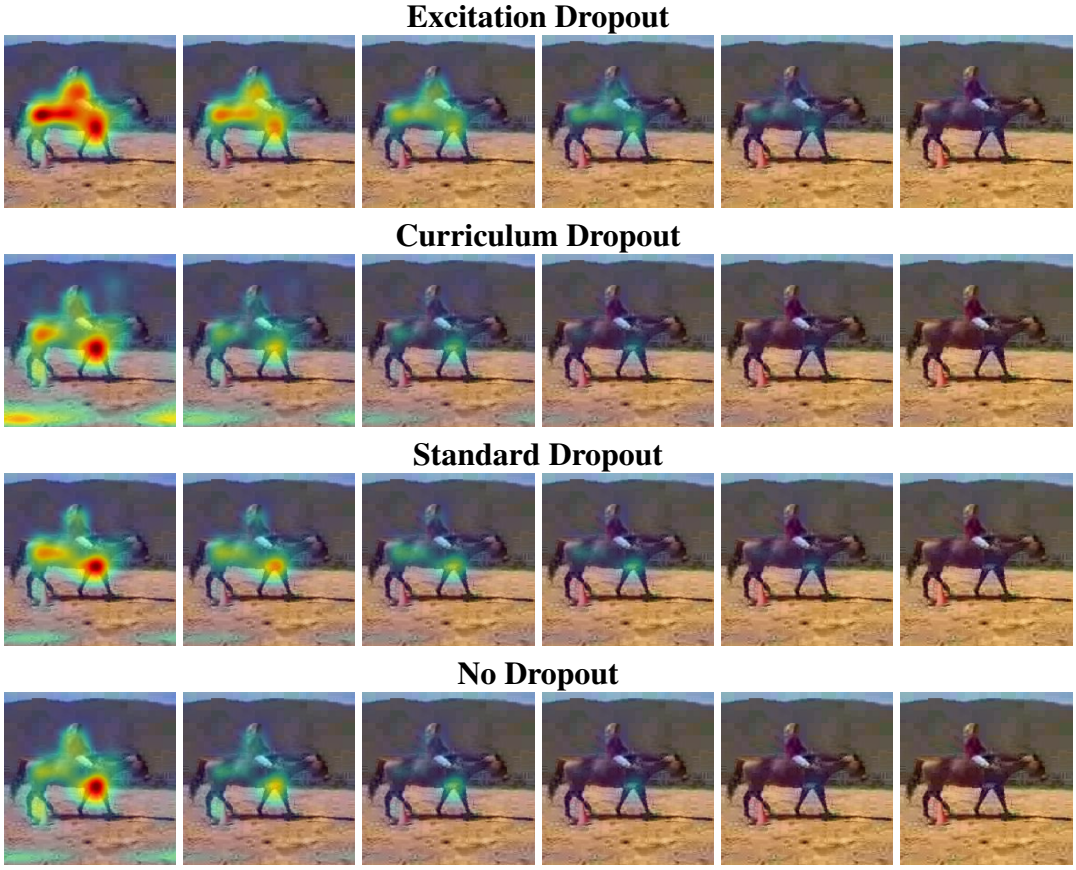


Figure 5.4: Visualizations for a VGG16 network fine-tuned on UCF101. Every column displays the saliency map over the same video frame of the action *HorseRiding* while incrementally switching off the most k relevant/salient neurons ($k = 0, 100, 200, \ldots, 500$) in the *fc6* layer at test time. Excitation Dropout shows more robustness when more neurons are switched off. This is demonstrated through its ability to recover more of the saliency map even when a high percentage of the most salient neurons is dropped-out. This ability reflects the alternative learnt paths.

quantitative analysis on the entire test set after training is complete. We study how the predicted ground-truth (GT) probability changes as more neurons are dropped-out at test time. On the left we present the worst case when the neurons dropped are the most relevant to the prediction. The horizontal axis in the graph represents p_c, where $0 \leq p_c \leq 1$ is the cumulative sum of p_{EB} of the most 'important' neurons which will be switched off. The analysis is performed for $p_c = \{0, 0.05, \ldots, 0.90, 0.95\}$. In the center we present an

		Standard Dropout	Excitation Dropout
Run-time	**CNN-2** ***(1 iter)***	0.1532 ± 0.0064	0.1885 ± 0.0070 (+23%)
	VGG16 ***(1 iter)***	2.2928 ± 0.0297	2.8202 ± 0.0312 (+23%)

Table 5.4: Run-time comparison: Average time of 100 iterations (in seconds, batch size=50) for a Caffe python layer on a GTX Titan X GPU and Intel(R) Xeon(R) CPU E5-2650 v3 @ 2.30GHz. In parenthesis is the percentage increase with respect to SD.

analogous analysis dropping the 'least' relevant neurons. On the right, we present the random case (more realistic) when k neurons ($k = 0, 128, 256, \ldots, 4096$) are randomly switched off. As we drop more neurons, ED (purple curves) is capable of maintaining a much less steep decline of GT probability, indicating more robustness against network compression.

5.2.5 Analysis

In this section we analyze the model complexity of Excitation Dropout, perform a sensitivity analysis over the dropout rate hyper-parameter, and extend the analysis of the network utilization metrics over the training iterations.

Model complexity. Excitation Dropout consistently outperforms Standard Dropout (SD), with zero increase in test-time computational complexity. In training, there is a moderate increase in computation: in the worst case, ED will take double (same O-notation complexity) the training time of SD. This will happen when the utilized ED maps are at the first layer of the network. If a middle layer map is used, ED requires an additional ***partial*** forward-backward pass. We use maps of fc layers close to the end of the network to reduce this overhead. Table 5.4 presents a run-time analysis for the two main architectures used in this chapter and compares it to that of Standard Dropout.

Sensitivity analysis. In this section we present a sensitivity analysis over the dropout rate hyper-parameter. We implement ED with a base retaining probability P, and we

Dataset	Dropout Scheme	0.25 Dropout	0.5 Dropout	0.75 Dropout
Cifar10	Standard	79.16%	80.13%	81.19%
	Excitation	**81.38%**	**81.94%**	**81.55%**
Cifar100	Standard	48.44%	50.36%	51.64%
	Excitation	**53.23%**	**52.04%**	**51.87%**
Caltech256	Standard	26.23%	28.73%	32.51%
	Excitation	**33.60%**	**35.77%**	**36.81%**
UCF101	Standard	71.01%	71.93%	72.92%
	Excitation	**73.56%**	**73.23%**	**73.06%**

Table 5.5: Hyper-parameter sensitivity analysis for the SD dropout probability, and the ED base dropout probability on the test set of different dataset. The retaining probability p or P is one minus the dropout rate.

compare that to standard dropout with a retaining probability p, where $P = p$. If ED produced a uniform probability distribution over the desired layer, then every node would have a retain probability equal to the base probability P. For completeness, we add a sensitivity analysis of the parameters p and P in Table 5.5.

Metric Analysis During Training. In this section we report an extended analysis of the metrics: *# Neurons ON*, *Peak* p_{EB}, *Entropy of Activations*, and *Entropy of* p_{EB} during training. Excitation Dropout shows a higher number of active neurons, a higher entropy over activations, a probability distribution p_{EB} that is more spread (higher entropy over p_{EB}) among the neurons of the layer, leading to a lower peak probability of p_{EB} and therefore less specialized neurons. These results are observed to have consistent trends over all training iterations for all datasets considered (see Figures 5.6, 5.7, 5.8, and 5.9).

5.3 Discussion

In this chapter, we demonstrate how grounding can be used at training time to improve network generalization both on shallow networks and on deeper networks for obtaining state-of-the-art results. We also demonstrate an increased utilization of network neurons

at training. In addition, we demonstrate how our approach increases network resilience to compression, a desired feature for having lighter, and therefore faster network for deployment on mobile devices. In the next chapter, we demonstrate how grounding can be used at test time to (a) question whether the evidence used to make a prediction is reasonable with respect to the evidence of correctly classified training examples, and (b) refine the prediction to one that has more reasonable evidence if necessary.

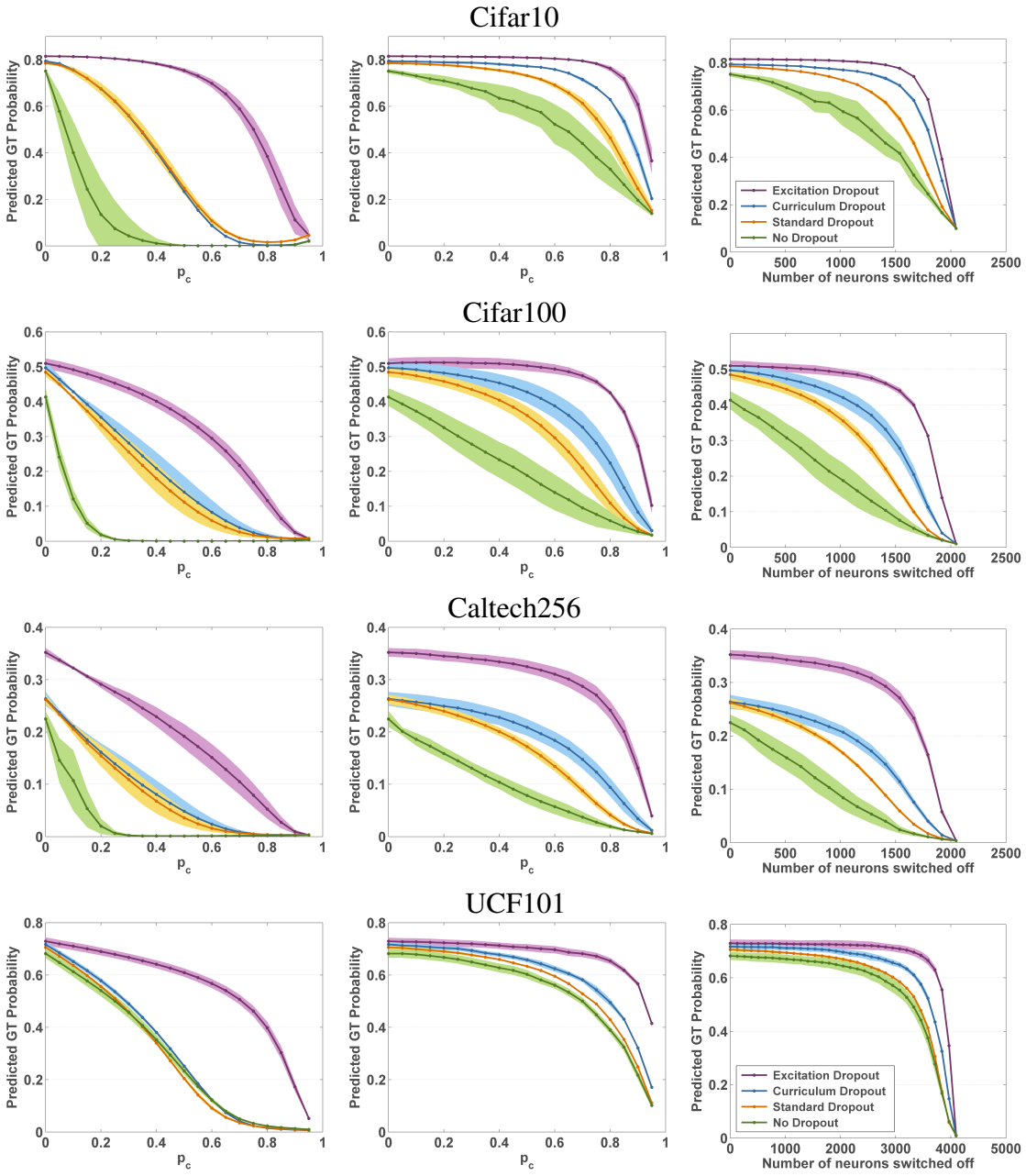


Figure 5.5: Robustness of predicted ground-truth class probabilities as more neurons are dropped-out for test images of the four benchmark datasets. We train CNN-2 from scratch (fine-tune VGG16 for UCF101) with Excitation, Curriculum, Standard, and No Dropout at the $fc1$ layer ($fc6$ for UCF101), averaging results over five trained models. At test time, we switch off the most relevant neurons with respect to p_c (left), the least relevant neurons with respect to p_c (center), and k random neurons (right). In all scenarios, Excitation Dropout shows more robustness to network compression (dropping fc neurons $\equiv$ removing filters).

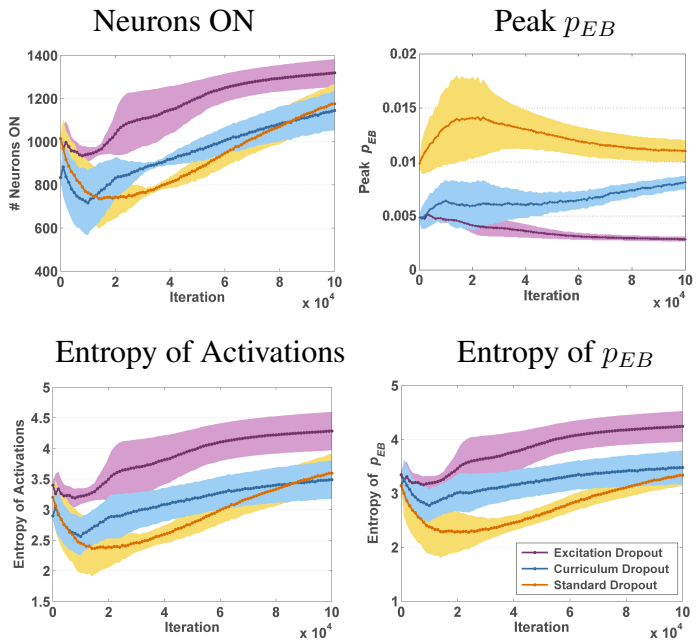


Figure 5.6: **Cifar10.** *# Neurons ON*, *Peak* p_{EB}, *Entropy of Activations*, and *Entropy of* p_{EB} over time during training.

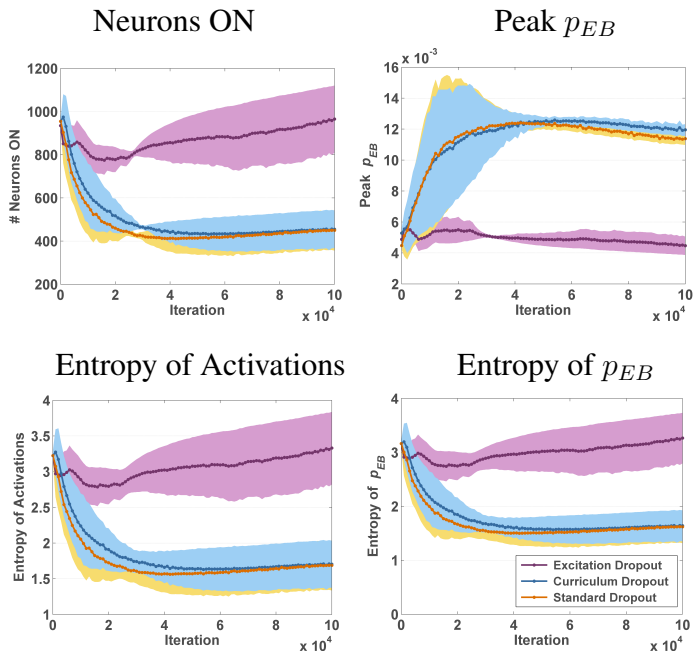


Figure 5.7: **Cifar100.** *# Neurons ON*, *Peak* p_{EB}, *Entropy of Activations*, and *Entropy of* p_{EB} over time during training.

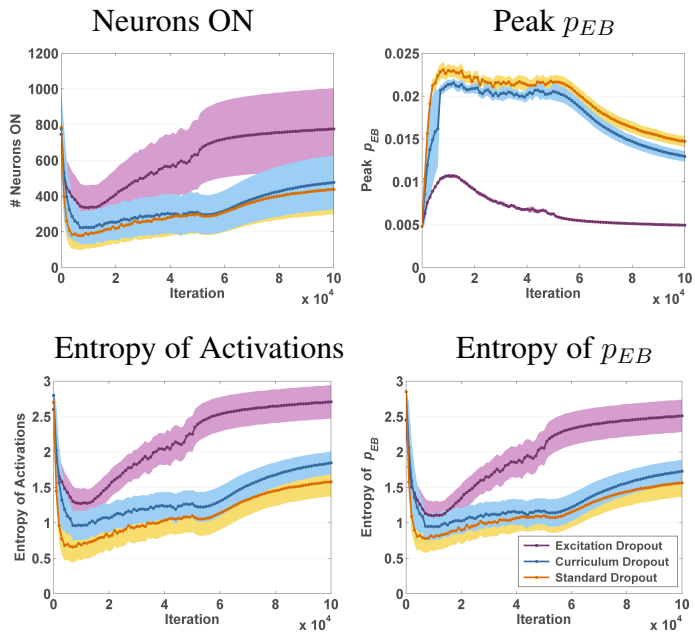


Figure 5.8: **Caltech256.** *# Neurons ON*, *Peak* p_{EB}, *Entropy of Activations*, and *Entropy of* p_{EB} over time during training.

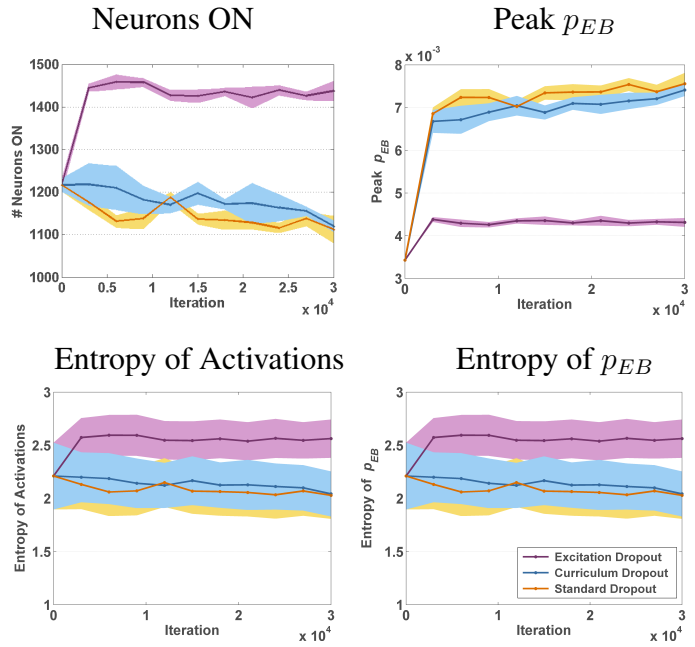


Figure 5.9: **UCF101.** *# Neurons ON*, *Peak* p_{EB}, *Entropy of Activations*, and *Entropy of* p_{EB} over time during training.

Chapter 6

`Guided Zoom`: Questioning Network Evidence for Fine-grained Classification

For state-of-the-art deep single-label classification models, the correct class is often in the top-k predictions, leading to a top-k ($k = 2, 3, 4, \dots$) accuracy that is significantly higher than the top-1 accuracy. This is also more pronounced in fine-grained classification tasks, where the differences between classes are quite subtle. For example, the Stanford Dogs fine-grained dataset on which we report results has a top-1 accuracy of 86.9% and a top-5 accuracy of 98.9%. Exploiting the information provided in the top k predicted classes can boost the final prediction of a model. In this chapter, we do not completely trust the model's top-1 prediction as it does not solely depend on the visual evidence in the input image, but can depend on other artifacts such as dataset bias or unbalanced training data. Instead, we exploit the discriminative visual evidence used for each of the top-k predictions for decision refinement.

Examples of fine-grained classes present in the literature are breeds of animals [57] and birds [134], models of aircraft [78] and vehicles [64]. Since fine-grained classification requires focusing on details, the localization of salient parts is crucial. This has been addressed using supervised approaches that utilize part bounding box annotations [146, 149, 41] or have humans in the loop to help reveal discriminative parts [21]. Part localization has also been addressed using weakly supervised approaches [31, 115, 154, 45], solely

relying on image labels during both training and testing. Another class of works attend to a recursively zoomed location [31, 81], while other methods use multiple attention mechanisms [115, 154]. Some approaches enforce correlations between parts [115, 45], while others do not consider this possible source of information [63, 31].

In this chapter, we want to answer the following question: *is the evidence upon which the prediction is made reasonable?* Evidence is defined to be the grounding, in pixel space, for a specific class conditional probability in the model output. The evidence proposed here is in the form of a saliency map resulting from weak supervision. It is directly obtained using grounding approaches that utilize a network's internal representation and a dataset's image-level annotation. We use evidence grounding as the signal to a module that assesses how much one can trust a Convolutional Neural Network (CNN) prediction over another.

We propose `Guided Zoom`, an approach that utilizes spatial grounding to refine model predictions in fine-grained classification scenarios. `Guided Zoom` zooms in on the evidence used to make a preliminary decision at test time and compares it with the evidence of correct predictions made at training time. As demonstrated in Figure 6.1, we propose not to solely rely on the prediction a conventional CNN produces, but to examine whether or not the evidence used to make the prediction is coherent with training evidence of correctly classified images. This is performed by the *Evidence CNN* module, which aids the *Decision Refinement* module to come up with a refined prediction. The desired goal in `Guided Zoom` is that the evidence of the refined class prediction is more coherent with the training evidence of that class, than the evidence of any of the other candidate top classes as depicted in Figure 6.2.

Our approach does not require part annotations, thus it is more scalable compared to supervised approaches. Moreover, our approach uses multiple salient regions and therefore does not propagate errors from an incorrect initial saliency localization, while implicitly

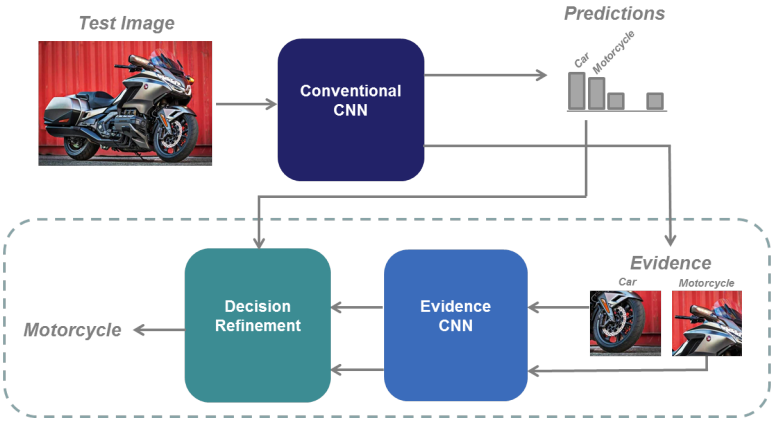


Figure 6.1: Pipeline of `Guided Zoom`. A conventional CNN outputs class conditional probabilities for an input image. Salient patches could reveal that evidence is weak. We refine the prediction of the conventional CNN by introducing two modules: 1) *Evidence CNN* determines the consistency between the evidence of a test image prediction and that of correctly classified training examples of the same class. 2) *Decision Refinement* uses the output of *Evidence CNN* to refine the prediction of the conventional CNN.

enforcing part correlations enabling models to make more informed predictions.

As the experiments of Wei *et al.* [132] suggest, although only part(s) of an object will be highlighted in the evidence, a more inclusive segmentation map can be extracted from the already trained model at test time. We follow their strategy of adversarial erasing to obtain a rich representation for the *Evidence CNN* module. We also investigate the complementarity of grounding techniques by comparing their ensemble performance to that of the adversarial erasing strategy. By questioning network evidence, we demonstrate refined accuracy on three fine-grained classification benchmark datasets.

6.1 Method

In this section, we describe the modules of our method depicted in Figure 6.1: *Evidence CNN* and *Decision Refinement*. Section 6.1.1 explains how we use the evidence of a prediction to improve classification performance by utilizing a pool of “reasonable" class

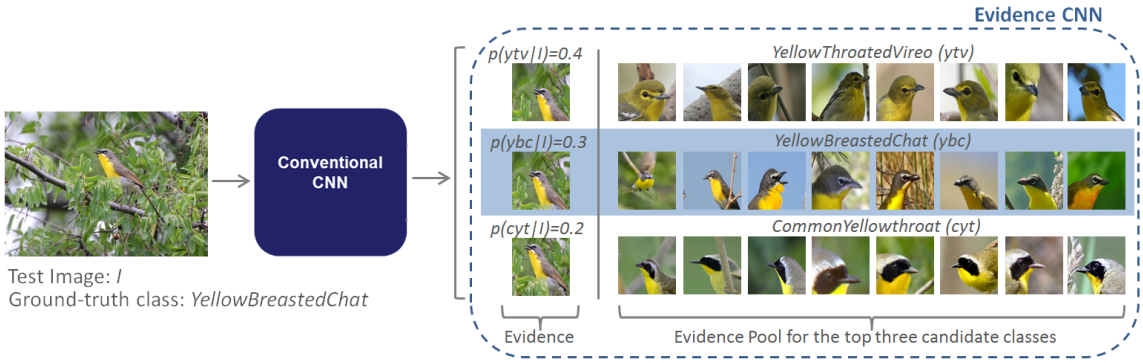


Figure 6.2: A conventional CNN could be used to obtain salient image regions that highlight the evidence for predictions, together with the predicted class conditional probabilities. Fine-grained classification decisions can be improved by comparing consistency of the evidence for the incoming test image with the evidence seen for correct classifications in training. In this demonstration, although the conventional CNN predicts with highest probability the class *YellowThroatedVireo*, the *Evidence CNN* is able to provide guidance for predicting the ground-truth class *YellowBreastedChat* (highlighted in blue) due to visual similarity of the evidence of this class with that of the pool of correctly classified training examples.

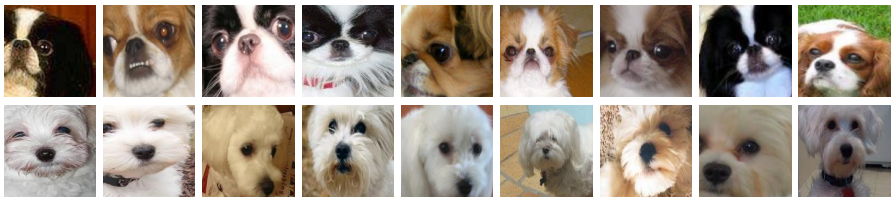
evidence, and Section 6.1.2 describes an alternative way to populate the evidence pool using different grounding techniques, exploring their complementarity.

6.1.1 Guided Zoom

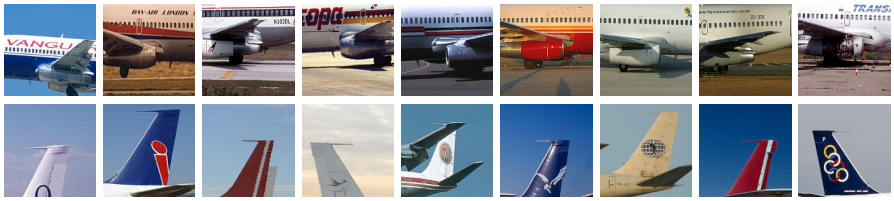
Evidence CNN. Conventional CNNs trained for image classification output class conditional probabilities upon which predictions are made. The class conditional probabilities are the result of some corresponding evidence in the input image. We recover/ground such evidence using spatial grounding methods, including contrastive Excitation Backprop (*c*EB) [147]. Starting with a prior probability distribution, *c*EB passes top-down signals through excitatory connections (having non-negative weights) of a CNN. Recursively propagating the top-down signal layer by layer, *c*EB computes class-specific discriminative saliency maps from any intermediate layer in a partial single backward pass.

We generate a reference pool, $\mathcal{P}$ of (evidence, prediction) pairs over which *Evidence*

(a) Sample discriminative patches for two classes of bird species: *RedWinged-Blackbird*, and *YellowHeadedBlackbird*



(b) Sample discriminative patches for two classes of dog species: *JapaneseS-paniel*, and *MalteseDog*



(c) Sample discriminative patches for two classes of aircraft models: *737-200*, and *707-320*

Figure 6.3: Most salient patches extracted from the conventional CNN using the spatial grounding approach contrastive Excitation Backprop (*c*EB). Such patches are then used to train the *Evidence CNN* to differentiate zoomed in details for each class. Patches of different images are presented from two sample classes of the fine-grained datasets (a) CUB-201-2011 Birds, (b) Stanford Dogs, and (c) FGVC-Aircraft.

CNN will be trained for the same classification task. Pairs in the pool $\mathcal{P}$ are extracted for correctly classified training examples using the grounding method *c*EB. This is done by setting the prior distribution in correspondence with the correct class to produce a *c*EB saliency map for it. We extract 150x150-pixel patches from the original image around the resulting peak saliency. Such patches are demonstrated in Figure 6.3 for fine-grained datasets of birds, dogs, and aircraft. The patches highlight the most discriminative evidence for two sample classes of each dataset. For example, the most discriminative evidence to

Figure 6.4: Implicit part detection obtained as a result of two iterations of adversarial erasing. The first row shows the most salient patches of four images from the class *Chihuahua* in the Stanford Dogs dataset. The second row shows the second most salient patches, and the third row shows the third most salient patches for the same four images. Assigning the same class label to the different parts of a single dog image enforces implicit part-label correlation.

differentiate dogs tends to be the face. However, the next most discriminative patches may also be good additional evidence for differentiating fine-grained categories.

Inspired by the adversarial erasing work of Wei *et al.* [132], we augment our reference pool with patches resulting from performing an iterative adversarial erasing of the most discriminative evidence from the image. We notice that adversarial erasing results in implicit part localization from the most to least discriminative parts. Figure 6.4 shows the patches extracted from two iterations of adversarial saliency erasing for sample images belonging to the class *Chihuahua* from the Stanford Dogs Dataset. All patches (parts) extracted from this process inherit the ground-truth label of the original image. By labeling different parts with the same image ground-truth label, we are implicitly forcing part-label correlations in *Evidence CNN*.

Including such additional evidence in our reference pool gives a richer description of the examined classes compared to models that recursively zoom into one location and ignore

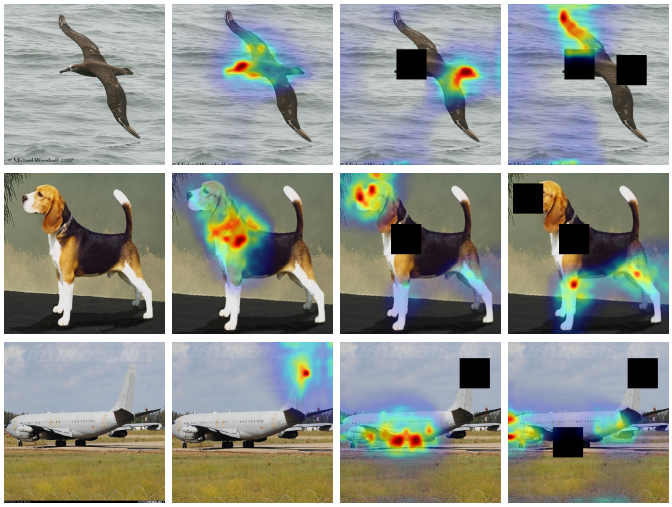


Figure 6.5: Sample image from each dataset to demonstrate the extraction of patches during two rounds of adversarial erasing: finding the first ($l = 0$), second ($l = 1$), and third ($l = 2$) most-salient evidence for a *BlackFootedAlbatross* bird, an *EnglishFoxhound* dog, and a *707-320* aircraft. For example, the most salient evidence for the bird image is the head, followed by the tail, followed by the right wing.

the less discriminative cues [31]. We note that we add an evidence patch to the reference pool only if the removal of previous salient patch does not affect the correct classification of the sample s^i. Erasing is performed by adding a black-filled 85x85-pixel square on the previous most salient evidence to encourage a highlight of the next most salient evidence. This process is depicted in Figure 6.5 for a sample bird species, dog species, and aircraft model.

Assuming n training samples, for each sample s^i where $i \in 1, \ldots, n$ we have $l + 1$ evidence patches in the reference pool $e^i_0, \ldots, e^i_l$. e^i_0 is the most discriminative initial evidence, and $e^i_1, \ldots, e^i_l$ is the set of l next discriminative evidence where $l \leq L$ and L is the number of adversarial erasing iterations performed ($L = 2$ is used in our experiments). For example, e^i_2 is the third most-discriminative evidence, after the erasing of e^i_0 and e^i_1 from the original image. Construction of the reference pool is summarized in Algorithm 1.

Algorithm 1: Generation of Evidence Pool $\mathcal{P}$

Input: $s^i, i \in 1, \ldots, n$ training images, pre-trained conventional CNN, Grounding Method (GM)

Output: Evidence Pool $\mathcal{P}$

Procedure:

Initialize evidence pool $\mathcal{P} = \{\}$
For every training example $s^i \in 1, \ldots, n$
  If s^i is correctly classified by conventional CNN
    Compute $e_0^i :=$ GM(s^i) w.r.t. ground-truth class
    $\mathcal{P} = \mathcal{P} \cup e_0^i$
    For $l \in 1, \ldots, L$
      Adversarially erase e_{l-1}^i from s^i
      If s^i is correctly classified by conventional CNN
        Compute next-salient patch for s^i: $e_l^i =$ GM(s^i)
        $\mathcal{P} = \mathcal{P} \cup e_l^i$

We then train a CNN model, *Evidence CNN*, on the generated evidence pool $\mathcal{P}$.

Decision Refinement. At test time, we analyze whether the evidence upon which a prediction is made is reasonable. We do so by examining the consistency of a test (evidence, prediction) with our reference pool that is used to train *Evidence CNN*. The refined prediction will be biased toward each of the top-k classes by an amount proportional to how coherent its evidence is with the reference pool. For example, if the (evidence, prediction) of the second-top predicted class is more coherent with the reference pool of this class, then the refined prediction will be more biased toward the second-top class.

Assuming test image s^j, where $j \in 1, \ldots, m$ and m is the number of testing examples, s^j is passed through the conventional CNN resulting in $v^{j,0}$, a vector of class conditional probabilities having some top-k classes $c_1, \ldots, c_k$ to be considered for the prediction refinement. We obtain the evidence for each of the top-k predicted classes $e_0^{j,c_1}, \ldots, e_0^{j,c_k}$,

Algorithm 2: Decision Refinement

Input: $s^j, j \in 1, \ldots, m$ testing images, pre-trained conventional CNN, pre-trained *Evidence CNN*, Grounding Method (GM), $w, w_0, \ldots, w_L$

Output: Refined class for s^j: c^j_{ref}

Procedure:

For every test example $s^j \in 1, \ldots, m$
  $v^{j,0}$:= conventional CNN(s^j)
  tot^j:= $w * v^{j,0}$
  For $t \in c_1, \ldots, c_k$, the top-k classes of $v^{j,0}$
    $e^{j,t}_0$:= GM(s^j) w.r.t. class t
    $v^{j,t}_0$:= *Evidence CNN*($e^{j,t}_0$)
    $tot^j[t] := tot^j[t] + w_0 * v^{j,t}_0[t]$
    For $l \in 1, \ldots, L$
      Adversarially erase e^i_{l-1} from s^i
      $e^{j,t}_l$:= GM(s^j) w.r.t. class t
      $v^{j,t}_l$:= *Evidence CNN*($e^{j,t}_l$)
      $tot^j[t] := tot^j[t] + w_l * v^{j,t}_l[t]$
  $c^j_{ref} := \underset{c_1:c_k}{\operatorname{argmax}}(tot^j)$

and pass each one through the *Evidence CNN* to get the following output class conditional probability vectors $v^{j,c_1}_0, \ldots, v^{j,c_k}_0$. We then perform adversarial erasing to get the next most salient evidence $e^{j,c_1}_l, \ldots, e^{j,c_k}_l$ and their corresponding class conditional probability vectors $v^{j,c_1}_l, \ldots, v^{j,c_k}_l$, for $l \in 1, \ldots, L$. Finally, we compute a weighted combination of the class conditional probability vectors proportional to their saliency. The estimated, refined class c^j_{ref} is determined as the class having the maximum aggregate prediction in the weighted combination. Algorithm 2 presents the steps used for decision refinement.

6.1.2 Ensemble Guided Zoom

We explore the utilization of an ensemble of evidence grounding techniques [147, 103, 93] to investigate whether their complementarity could be comparable to the explicit adversarial erasing of salient regions in the evidence pool generation process explained in Section 6.1.1. We use saliency maps from contrastive Excitation Backprop (cEB) [147], Gradient-weighted Class Activation Mapping (Grad-CAM) [103], and Randomized Input Sampling for Explanation (RISE) [93]. Equation 6.1 presents the proposed augmented evidence pool.

$$\mathcal{P} = \mathcal{P}_{cEB} \cup \mathcal{P}_{Grad\text{-}CAM} \cup \mathcal{P}_{RISE} \tag{6.1}$$

$\mathcal{P}_{cEB}$, $\mathcal{P}_{Grad\text{-}CAM}$, and $\mathcal{P}_{RISE}$ are each generated following Algorithm 1 using $L = 0$ (without adversarial erasing) and using cEB, Grad-CAM, or RISE for the grounding method, respectively.

cEB is a discriminative top-down saliency approach that is probabilistically interpretable. It cancels out the common winner neurons and amplifies the class discriminative neurons. We compute each saliency map using a partial backward pass of the pre-trained conventional CNN and populate $\mathcal{P}_{cEB}$ accordingly.

Grad-CAM is a class-discriminative localization technique, that also requires a partial backward pass of the pre-trained conventional CNN. We use Grad-CAM to compute saliency maps for populating $\mathcal{P}_{Grad\text{-}CAM}$.

RISE randomly samples masks for the input image, and based on the respective change in the predicted class conditional probabilities, aggregates such masks to produce a saliency map without using any model parameters. We use the pre-trained conventional CNN as a black-box model, and compute saliency maps to populate $\mathcal{P}_{RISE}$.

Saliency maps from cEB, Grad-CAM, and RISE are used to extract 150x150-pixel evidence patches from the corresponding original image around the peak saliency. Such

original *c*EB RISE Grad-CAM

Figure 6.6: Sample saliency images produced by *c*EB, RISE, and Grad-CAM for a *CrestedAuklet* bird, an *AfghanHound* dog, and an *A318* aircraft. It is interesting to observe some complementarity as in adversarial erasing.

patches are used with their corresponding image-level class label as (evidence, prediction) pairs to train the *Evidence CNN*. Figure 6.6 depicts sample saliency maps produced by the three spatial grounding techniques for fine-grained datasets of bird species, dog species, and aircraft models. We observe some complementarity in the aggregation of these salient regions, as grounding techniques do not consistently highlight the same image regions as evidence for a specific class. Results for both `Guided Zoom` and `Ensemble Guided Zoom` are presented in the next section.

6.2 Experiments

In this section, we first present the fine-grained benchmark datasets we use to evaluate `Guided Zoom` and `Ensemble Guided Zoom`. We then present the architecture and setup of our experiments, followed by a discussion of our experimental results. We note

that although the datasets provide part annotations, we only use image-level class labels.

Datasets. We report experimental results on three fine-grained classification benchmark datasets following [115, 31, 152, 19, 154].

- CaltechUCSD (CUB-200-2011) Birds Dataset [134] is a fine-grained dataset of 200 bird species consisting of ∼12K annotated images, split into ∼6K training images and ∼6K testing images.

- Stanford Dogs Dataset [57] is a fine-grained dataset of 120 dog species. This dataset includes ∼20K annotated images split into ∼12K and ∼8.5K images for training and testing respectively.

- FGVC-Aircraft [78] is a fine-grained dataset of 100 different aircraft variants consisting of 10K annotated images, split into ∼7K training images and ∼3K testing images.

Architecture and Setup. To validate the benefit of `Guided Zoom`, we purposely use a simple CNN baseline with a vanilla training scheme. We use a ResNet-101 [37] network as the conventional CNN and baseline, extending the input size from the default 224x224-pixel to 448x448-pixel following [115, 31, 63]. The 448x448-pixel input image is a random crop from a 475x475-pixel input image at training time, and a center crop from a 475x475-pixel input image at test time.

For the *Evidence CNN*, we use a ResNet-101 architecture, but use the standard 224x224-pixel input size to keep the patches close to their original image resolution. This is a random crop from a 256x256-pixel input image at training time, and a center crop from a 256x256-pixel input image at test time. For both the conventional and *Evidence* CNNs, and for all the three datasets, we use stochastic gradient descent, a batch size of 64, a

	Method	Part / Whole Annotation	Multiple Attention	Top-1 Accuracy (%)
	DVAN [152]	x	✓	79.0
	PA-CNN [62]	✓	✓	82.8
	MG-CNN [122]	✓	✓	83.0
	B-CNN [73]	x	x	84.1
	RA-CNN [31]	x	x	85.3
	PN-CNN [13]	✓	✓	85.4
	OSME + MAMC [115]	x	✓	**86.5**
	MA-CNN [154]	x	✓	**86.5**
	ResNet-101 Baseline	x	x	82.3
Ours	`Guided Zoom` (k=3)	x	✓	85.0
	`Guided Zoom` (k=5)	x	✓	85.4
	`Ensemble Guided Zoom` (k=3)	x	✓	84.6
	`Ensemble Guided Zoom` (k=5)	x	✓	85.0

Table 6.1: **CUB-200-2011 Birds Dataset.** We compare our classification accuracy with state-of-the-art weakly-supervised methods (do not use any sort of annotation apart from the image label) and some representative methods that use additional supervision such as part annotations for fine-grained classification of this dataset. We indicate which methods use multiple parts, and which focus on a single part using the multiple attention flag; using part annotations implicitly entails multiple attention. We present results for our approach for k=3,5; using the top 3 (or 5) candidate classes to refine the final prediction.

starting learning rate of 0.001, multiplied by 0.1 every 10K iterations for 30K iterations, and momentum of 0.9.

We demonstrate the benefit of using evidence information from the top-3 and top-5 predicted classes, so we set $k = 3, 5$ in our experiments. We perform two rounds of adversarial erasing in testing; setting $L = 2$, $w = 0.4, w_0 = 0.3, w_1 = 0.2$, and $w_2 = 0.1$.

Results. We now present results on the three fine-grained datasets: CUB-200-2011 Birds, Stanford Dogs, and FGVC-Aircraft. In this section, we demonstrate how training our *Evidence CNN* benefits from (a) using implicit part detection by adversarial erasing to obtain the next most-salient evidence, and (b) using an ensemble of evidence grounding techniques, both of which target providing complementary zooming on salient parts.

For the CUB-200-2011 Birds dataset, our conventional CNN (ResNet-101 baseline)

	Method	Part / Whole Annotation	Multiple Attention	Top-1 Accuracy (%)
	DVAN [152]	x	✓	81.5
	OSME + MAMC [115]	x	✓	85.2
	RA-CNN [31]	x	x	87.3
	ResNet-101 Baseline	x	x	86.9
Ours	`Guided Zoom` (k=3)	x	✓	88.4
Ours	`Guided Zoom` (k=5)	x	✓	**88.5**
Ours	`Ensemble Guided Zoom` (k=3)	x	✓	88.3
Ours	`Ensemble Guided Zoom` (k=5)	x	✓	88.3

Table 6.2: **Stanford Dogs Dataset.** We compare our classification accuracy with state-of-the-art weakly-supervised methods (do not use any sort of annotation apart from the image label). We indicate which methods use multiple parts, and which focus on a single part using the multiple attention flag; using part annotations implicitly entails multiple attention. We present results for our approach for k=3,5; using the top 3 (or 5) candidate classes to refine the final prediction.

achieves 82.3% top-1 accuracy, 92.8% top-3 accuracy, and 95.6% top-5 accuracy. Table 6.1 presents the results for the CUB-200-2011 Birds dataset. Utilizing the top-3 class predictions together with their associated evidence, `Guided Zoom` boosts the top-1 accuracy from 82.3% to 85.0%, while `Ensemble Guided Zoom` boosts the top-1 accuracy from 82.3% to 84.6%. Utilizing the top-5 class predictions together with their associated evidence, `Guided Zoom` boosts the top-1 accuracy from 82.3% to 85.4%, while `Ensemble Guided Zoom` boosts the top-1 accuracy from 82.3% to 85.0%.

For the Stanford Dogs dataset, our conventional CNN (ResNet-101 baseline) achieves 86.9% top-1 accuracy, 97.8% top-3 accuracy, and 98.9% top-5 accuracy. Table 6.2 presents the results for the Stanford Dogs dataset on which `Guided Zoom` obtains state-of-the-art results. Utilizing the top-3 class predictions together with their associated evidence, `Guided Zoom` boosts the top-1 accuracy from 86.9% to 88.4%, while `Ensemble Guided Zoom` boosts the top-1 accuracy from 86.9% to 88.3%. Utilizing the top-5 class predictions together with their associated evidence, `Guided Zoom` boosts the top-

	Method	Part / Whole Annotation	Multiple Attention	Top-1 Accuracy (%)
	B-CNN [73]	x	x	84.1
	MG-CNN [122]	✓	✓	86.6
	RA-CNN [31]	x	x	88.2
	MDTP [130]	✓	✓	88.4
	MA-CNN [154]	x	✓	**89.9**
	ResNet-101 Baseline	x	x	87.5
Ours	`Guided Zoom` (k=3)	x	✓	89.1
	`Guided Zoom` (k=5)	x	✓	89.0
	`Ensemble Guided Zoom` (k=3)	x	✓	89.0
	`Ensemble Guided Zoom` (k=5)	x	✓	88.9

Table 6.3: **FGVC-Aircraft Dataset.** We compare our classification accuracy with state-of-the-art weakly-supervised methods (do not use any sort of annotation apart from the image label) and some representative methods that use additional supervision such as part annotations for fine-grained classification of this dataset. We indicate which methods use multiple parts, and which focus on a single part using the multiple attention flag; using part annotations implicitly entails multiple attention. We present results for our approach for k=3,5; using the top 3 (or 5) candidate classes to refine the final prediction.

1 accuracy from 86.9% to 88.5%, while `Ensemble Guided Zoom` boosts the top-1 accuracy from 86.9% to 88.3%.

For the FGVC-Aircraft dataset, our conventional CNN (ResNet-101 baseline) achieves 87.5% top-1 accuracy, 95.2% top-3 accuracy, and 96.1% top-5 accuracy. Table 6.3 presents the results for the FGVC-Aircraft dataset. Utilizing the top-3 class predictions together with their associated evidence, `Guided Zoom` boosts the top-1 accuracy from 87.5% to 89.1%, while `Ensemble Guided Zoom` boosts the top-1 accuracy from 87.5% to 89.0%. Utilizing the top-5 class predictions together with their associated evidence, `Guided Zoom` boosts the top-1 accuracy from 87.5% to 89.0%, while `Ensemble Guided Zoom` boosts the top-1 accuracy from 87.5% to 88.9%.

`Guided Zoom` outperforms RA-CNN on all three datasets. From this we can conclude that our multi-zooming is more beneficial than a single recursive zoom. `Guided Zoom`

outperforms OSME + MAMC on the Stanford Dogs Dataset, but the opposite is true for the CUB-200-2011 Birds Dataset. Being a generic framework, `Guided Zoom` could be used to further boost performance of state-of-the-art methods on the CUB-200-2011 Birds and FGVC-Aircraft datasets.

`Guided Zoom` uses *c*EB with adversarial erasing, while `Ensemble Guided Zoom` uses evidence from several grounding techniques. In Tables 6.1, 6.2, and 6.3, comparable results for `Guided Zoom` and `Ensemble Guided Zoom` indicate similar complementarity of object parts in both pool generation approaches, as initially demonstrated in Figure 6.5 and Figure 6.6.

6.3 Discussion

In this chapter, we devise a methodology that utilizes explicit spatial grounding to refine a model's prediction at test time. In the next chapter we address a much less explored avenue: weakly-supervised saliency for spatiotemporal architectures. We propose the first top-down saliency in deep recurrent models for space-time grounding of videos using a single *contrastive* Excitation Backprop pass of an already trained model. We demonstrate that such grounding can be used for coarse spatial and temporal localization of actions in video.

Chapter 7

Excitation Backprop for RNNs

To visualize what in a video gives rise to an output of a deep recurrent network, it is important to consider space and time saliency, *i.e.* , where and when. The visualization of what a deep recurrent network finds salient in an input video can enable interpretation of the model's behavior in action classification, video captioning, and other tasks. Moreover, estimates of the model's attention (*e.g.* , saliency maps) can be used directly in localizing a given action within a video or in localizing the portions of a video that correspond to a particular concept within a caption. In this chapter, we devise a formulation for top-down attention in recurrent neural network models for spatiotemporal grounding of visual data.

Several works address visualization of model attention in Convolutional Neural Networks (CNNs) for image classification [15, 148, 111, 145, 106, 156, 103]. These methods produce saliency maps that visualize the importance of class-specific image regions (spatial localization). Analogous methods for Recurrent Neural Network (RNN)-based models must handle more complex recurrent, non-linear, spatiotemporal dependencies; thus, progress on RNNs has been limited to [54, 97]. Karpathy *et al.* [54] visualize the role of Long Short Term Memory (LSTM) cells for text input, but not for visual data. Ramanishka *et al.* [97] map words to regions in the video captioning task by dropping out (exhaustively or by sampling) video frames and/or parts of video frames to obtain saliency maps. This can be computationally expensive, and does not consider temporal evolution but only frame-level

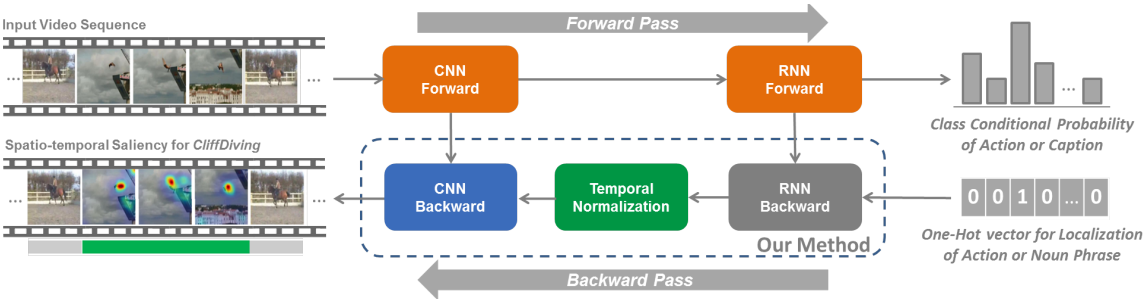


Figure 7.1: Our proposed framework spatiotemporally highlights/grounds the evidence that an RNN model used in producing a class label or caption for a given input video. In this example, by using our proposed back-propagation method, the evidence for the activity class *CliffDiving* is highlighted in a video that contains *CliffDiving* and *HorseRiding*. Our model employs a single backward pass to produce saliency maps that highlight the evidence that a given RNN used in generating its outputs.

saliency.

In contrast, we propose the first one-pass formulation for visualizing spatiotemporal attention in RNNs, without selectively dropping or sampling frames or frame regions. In our proposed approach, *contrastive* Excitation Backprop for RNNs (cEB-R), we address how to ground[1] decisions of deep recurrent networks in space and time simultaneously, using top-down saliency. Our approach models the top-down attention mechanism of deep models to produce interpretable and useful task-relevant saliency maps. Our saliency maps are obtained implicitly without the need to re-train models, unlike models that include explicit attention layers [139, 142]. Our method does not require a model trained using explicit spatial (region/bounding box) or temporal (frame) supervision.

Figure 7.1 gives an overview of our approach that produces saliency maps which enable us to visualize where and when an action/caption is occurring in a video. Given a trained model, we perform the standard forward pass. In the backward pass, we use cEB-R to compute and propagate winning neuron probabilities normalized over space and time. This

[1] In this work we use the terms *ground* and *localize* interchangeably.

process yields spatiotemporal attention maps. Our demo code is publicly available [2].

We evaluate our approach on two models from the literature: a CNN-LSTM trained for video action recognition, and a CNN-LSTM-LSTM (encoder-decoder) trained for video captioning. In addition, we show how the spatiotemporal saliency maps produced for these two models can be utilized for localization of segments within a video that correspond to specified activity classes or noun phrases.

In summary, our contributions are:

- We are the first to formulate top-down saliency in deep recurrent models for space-time grounding of videos.
- We do so using a *single contrastive* Excitation Backprop pass of an already trained model.
- Although we are not directly optimizing for localization (no training is performed on spatial or temporal annotations), we show that the internal representation of the model can be utilized to perform localization.

7.1 Method

In this section we explain the details of our spatiotemporal grounding framework: cEB-R. As illustrated in Figure 7.1, we have three main modules: RNN Backward, Temporal normalization, and CNN Backward.

RNN Backward. This module implements an excitation backprop formulation for RNNs. Recurrent models such as LSTMs are well-suited for top-down temporal saliency as they explicitly propagate information over time. The extension of EB for Recurrent Networks, EB-R, is not straightforward since EB must be implemented through the unrolled time steps of the RNN and since the original RNN formulation contains $tanh$

[2] https://github.com/sbargal/Caffe-ExcitationBP-RNNs

non-linearities which do not satisfy the EB assumptions **A1** and **A2**. [33, 51] have conducted an analysis over variations of the standard RNN formulation, and discovered that different non-linearities performed similarly for a variety of tasks. This is also reflected in our experiments. Based on this, we use *ReLU* nonlinearities and corresponding derivatives, instead of $tanh$. This satisfies **A1** and **A2**, and gives similar performance on both tasks.

Working backwards from the RNN's output layer, we compute the conditional winning probabilities from the set of output nodes O, and the set of dual output nodes $\overline{O}$:

$$P^t(a_i|a_j) = \begin{cases} Z_j \widehat{a}_i^t w_{ij}, & \text{if } w_{ij} \geq 0, \\ 0, & \text{otherwise.} \end{cases} \tag{7.1}$$

$$\overline{P}^t(a_i|a_j) = \begin{cases} Z_j \widehat{a}_i^t \overline{w}_{ij}, & \text{if } \overline{w}_{ij} \geq 0, \\ 0, & \text{otherwise.} \end{cases} \tag{7.2}$$

$Z_j = 1/\sum_{i:w_{ij}\geq 0} \hat{a}_i^t w_{ij}$ is a normalization factor such that the sum of all conditional probabilities of the children of a_j (Equations 7.1, 7.2) sum to 1; $w_{ij} \in W$ where W is the set of model weights and w_{ij} is the weight between child neuron a_i and parent neuron a_j; $\overline{w}_{ij} \in \overline{W}$ where $\overline{W}$ is obtained by negating the model weights at the classification layer only. $\overline{P}^t(a_i|a_j)$ is only needed for *contrastive* attention.

We compute the neuron winning probabilities starting from the prior distribution encoding a given action/caption as follows:

$$P^t(a_i) = \sum_{a_j \in \mathcal{P}_i} P^t(a_i|a_j) P^t(a_j) \tag{7.3}$$

$$\overline{P}^t(a_i) = \sum_{a_j \in \mathcal{P}_i} \overline{P}^t(a_i|a_j) \overline{P}^t(a_j) \tag{7.4}$$

where $\mathcal{P}_i$ is the set of parent neurons of a_i.

Temporal Normalization. Replacing $tanh$ non-linearities with *ReLU* non-linearities to extend EB in time does not suffice for temporal saliency. EB performs normalization at every layer to maintain a probability distribution. Hence, for spatiotemporal localization, signals from the desired n^{th} time-step of a T-frame clip should be normalized in both time and space (assuming S neurons in current layer) before being further backpropagated into the CNN:

$$P_N^t(a_i) = P^t(a_i) / \sum_{t=1}^{T} \sum_{i=1}^{S} P^t(a_i). \tag{7.5}$$

$$\overline{P}_N^t(a_i) = \overline{P}^t(a_i) / \sum_{t=1}^{T} \sum_{i=1}^{S} \overline{P}^t(a_i). \tag{7.6}$$

cEB-R computes the difference between the normalized saliency maps obtained by EB-R starting from O, and EB-R starting from $\overline{O}$ using negated weights of the classification layer. cEB-R is more discriminative as it grounds the evidence that is unique to a selected class/word. For example, cEB-R of *Surfing* will give evidence that is unique to *Surfing* and not common to other classes used at training time (see Figure 7.4 for an example). This is conducted as follows:

$$Map^t(a_i) = P_N^t(a_i) - \overline{P}_N^t(a_i). \tag{7.7}$$

CNN Backward. For every video frame f_t at time step t, we use the backprop of [148] for all CNN layers:

$$P^t(a_i|a_j) = \begin{cases} Z_j \widehat{a}_i^t w_{ij}, & \text{if } w_{ij} \geq 0, \\ 0, & \text{otherwise} \end{cases} \tag{7.8}$$

$$Map^t(a_i) = \sum_{a_j \in P_i} P^t(a_i|a_j) Map^t(a_j) \tag{7.9}$$

where $\widehat{a}_i^t$ is the activation when frame f_t is passed through the CNN. Map^t at the desired CNN layer is the cEB-R saliency map for f_t. Computationally, the complexity of cEB-R

is on the order of a *single* backward pass. Note that for EB-R, $P_N^t(a_j)$ is used instead of $Map^t(a_j)$ in Equation 7.9. The general framework applied to both video action recognition and captioning is summarized in Algorithm 3. Details of each task are discussed in the following two sections.

Algorithm 3: cEB-R

Input: T-frame video clip, pre-trained CNN-LSTM model, $\mathcal{A}$: action or word to be localized in the video.

Output: Spatial saliency maps of $\mathcal{A}$: Map^t for $t = 1, ..., T$.

Procedure:

1 Set a one-hot vector according to the desired action class or caption word $\mathcal{A}$ at the desired n^{th} time-step;

2 Backprop the indicator vector through time and down to the *fc* CNN layer using EB-R obtaining a saliency map Map^t at every time step t;

3 Normalize the resulting frame-wise saliency maps over time such that $\sum_{t=1}^{T} Map^t = 1$;

4 Repeat the above steps, with negated weights at the top layer to get a second set of T saliency maps;

5 Contrastive Operation: Subtract the resulting maps at the *fc* CNN layer to yield cEB for each time step;

6 Continue EB through the CNN to the desired *conv* layer to obtain the spatial grounding;

7 The sum of each spatial saliency map over time can be used to perform temporal grounding for $\mathcal{A}$;

7.2 Grounding: Video Action Recognition

In this task, we ground the evidence of a specific action using a model trained on action recognition. The task takes as input a video sequence and the action ($\mathcal{A}$) to be localized, and outputs spatiotemporal saliency maps for this action in the video. We use the CNN-LSTM implementation of [26] with VGG-16 [108] for our action grounding in video. This encodes the temporal information intrinsically present in the actions we want to localize. The CNN is truncated at the *fc7* layer such that the *fc7* features of frames feed into the recurrent unit. We use a single LSTM layer.

Performing cEB-R results in a sequence of saliency maps Map^t for $t = 1, ..., T$ at *conv5* (various layers perform similarly [148]). These maps are then used to perform the temporal grounding for action $\mathcal{A}$. Localizing the action, entails the following sequence of steps. First, the sum of every saliency map is computed to give a vector $\mathcal{S} \in \mathbb{R}^T$. Second, we find an anchor map with the highest sum. Third, we extend a window around the anchor map in both directions in a greedy manner until a saliency map with a negative sum is found. A negative sum indicates that the map is less relevant to the action $\mathcal{A}$ under consideration. This allows us to determine the start and end points of the temporal grounding, $s_{\mathcal{A}}$ and $e_{\mathcal{A}}$ respectively. Figure 7.2 depicts the cEB-R pipeline for the task of action grounding.

7.3 Grounding: Video Captioning

In this task, we ground evidence of word(s) using a model trained on video captioning. The task takes as input a video and word(s) to be localized, and outputs spatiotemporal saliency maps corresponding to the query word(s). We use the captioning model of [119] to test our cEB-R approach. This model consists of a VGG-16, followed by a mean pooling of the VGG *fc7* features, followed by a two-layer LSTM. Figure 7.3 depicts cEB-R for caption

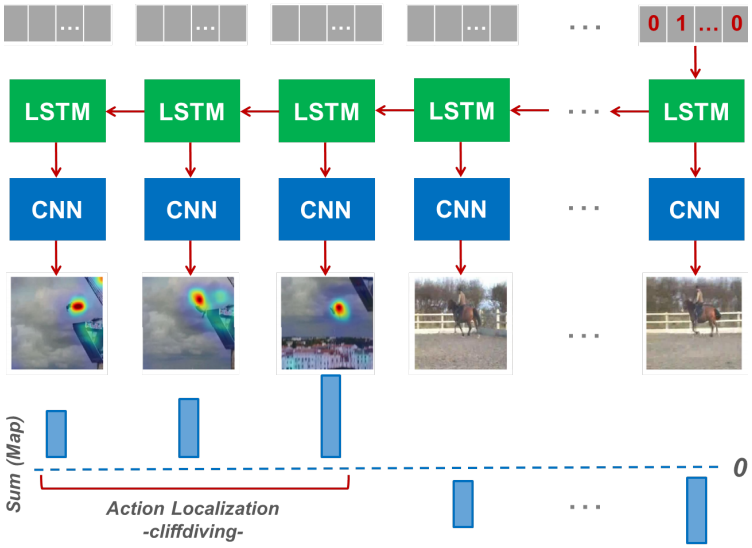


Figure 7.2: Grounding Action Recognition. The red arrows depict cEB-R for spatiotemporal grounding of the action *CliffDiving*. Starting from the last LSTM time-step, cEB-R backpropagates the probability distribution through time and through the CNN at every time-step. The saliency map for each time-step is used for the spatial localization. The sum of each saliency map, over time, is then used for temporal localization of the action within the video, as described in Sec. 7.2.

grounding.

We backpropagate an indicator vector for the words to be visualized starting at the time-steps they were predicted, through time, to the average pooling layer. We then distribute and backpropagate probabilities among frames -according to their forward activations (Equation 7.8)- through the VGG until the *conv5* layer where we obtain the corresponding saliency map. Performing cEB-R results in a sequence of saliency maps Map^t for $t = 1, ..., T$ grounding the words in the video frames. Temporal localization is performed using the steps described in Sec. 7.2.

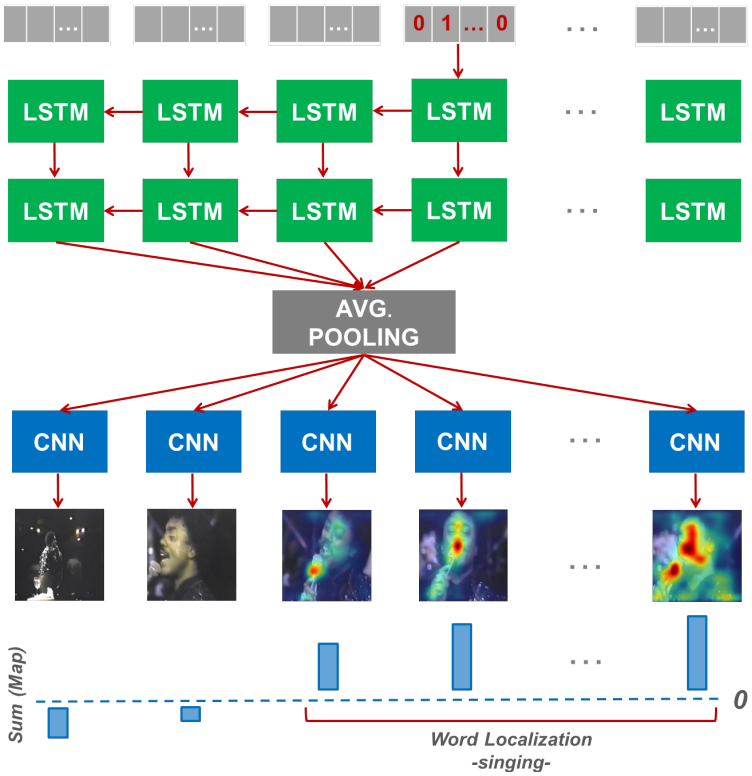


Figure 7.3: Grounding Captioning. The red arrows depict *c*EB-R for spatiotemporal caption grounding. The video caption produced by the model is *A man is singing on a stage*. Starting from the time-step corresponding to the word *singing*, *c*EB-R backprops the probability distribution through the previous time-steps and through the CNN. The saliency map for each time step is used for spatial localization. The sum of each saliency map, over time, is then used for temporal localization of the word within the clip.

7.4 Experiments: Action Grounding

In this work we ground the decisions made by our deep models. In order to evaluate this grounding, we compare it with methods that localize actions. Although our framework is able to jointly localize actions in space and time, we report results for spatial localization and temporal localization separately due to the lack of an action dataset that has untrimmed videos with spatiotemporal bounding boxes.

7.4.1 Spatial Localization

In this section we evaluate how well we ground actions in space. We do this by comparing our grounding results with ground-truth bounding boxes localizing actions per-frame.

Dataset. *THUMOS14* [50] provides per-frame bounding box annotations of humans performing actions for 3207 videos of 24 classes from the *UCF101* dataset [110]. *UCF101* is a trimmed video dataset containing 13320 actions belonging to 101 action classes.

Baselines. We compare our formulation against spatial top-down saliency using a CNN (treating every video frame as an independent image). We also compare against standard backpropagation (BP), and BP for RNNs (BP-R).

Models. We use the following CNN model: VGG-16 of Ma *et al.* [76] trained on *UCF101* video frames and BU101 web images for action recognition with a test accuracy of 83.5%. We use the following CNN-LSTM model: the same VGG-16 fine-tuned with a one-layer LSTM on *UCF101* for action recognition with a test accuracy of 83.3%.

Setup and Results. We use the bounding box annotations to evaluate our spatial grounding using the pointing game introduced by Zhang *et al.* [148]. We locate the point having maximum value on each top-down saliency map. Following [148], if a 15-pixel diameter circle around the located point intersects the ground-truth bounding-box of the action category for a frame, we record a hit, otherwise we record a miss. We measure the spatial action localization accuracy by $Acc = \#Hits/(\#Hits + \#Misses)$ over all the annotated frames for each action.

Table 7.1 reports the results of the spatial pointing game. Extending top-down saliency in time (-R) consistently improves the accuracy for all three methods, compared to performing top-down saliency separately on every frame of the video using a CNN. EB-R has the greatest absolute improvement of 5.7%.

Method Acc (%)					
EB	EB-R	cEB	cEB-R	BP	BP-R
55.8	**61.5**	37.0	**39.1**	37.3	**39.2**

Table 7.1: Accuracy of the spatial pointing game conducted on $\sim$3K videos of *UCF101* for spatially locating humans performing actions in videos. The results show that extending top-down saliency in time (-R) improves the accuracy compared to performing top-down saliency separately on every frame of the video using a CNN. The non-contrastive versions work better for reasons described in the text.

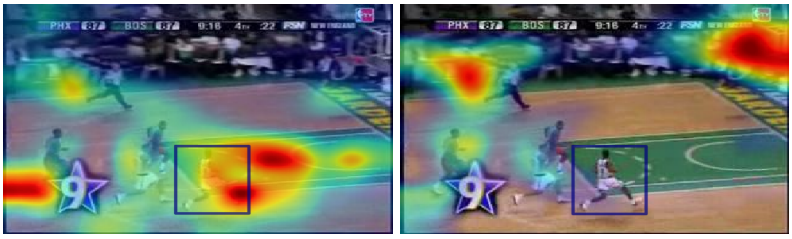
We note that the non-contrastive versions outperform their contrastive counterparts. This is because they highlight discriminative evidence for actions, which may not necessarily be the humans performing the actions. For example, for many actions in *UCF101*, the human may be in a standing position, in which case cEB-R will highlight cues that are discriminative and unique to this action rather than highlighting the human. These cues may belong to the context in which the activity is performed, or the action classes on which the model was trained. We demonstrate this in Figure 7.4 for the actions *Surfing* and *BasketballDunk*.

7.4.2 Temporal Localization

In this section we evaluate how well we ground actions in time. We do this by comparing our grounding results with ground-truth action boundaries.

Datasets. We first use a simple and controlled setting to validate our method by creating a synthetic action detection dataset. We then present results on the *THUMOS14* [50] action detection dataset. The synthetic dataset is created by concatenating two *UCF101* videos uniformly sampled: a ground truth (*GT*) video, and a random (*rand*) background video, such that class(*GT*) $\neq$ class(*rand*). The two actions are concatenated, first sequentially (*rand* + *GT* or *GT* + *rand*) in 16-frame clips, then inserted at a random position (*rand* + *GT* + *rand*) in 128-frame clips. We use all 3783 test videos provided in *UCF101*, each in

(a) Grounding *Surfing* using EB-R (L) and cEB-R (R)



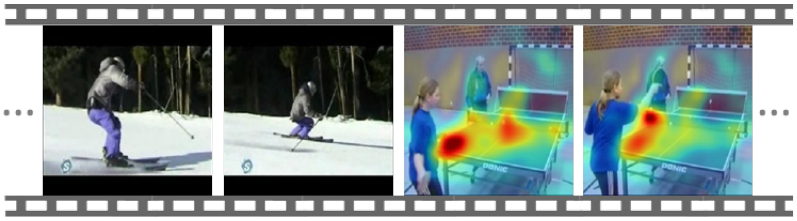
(b) Grounding *BasketballDunk* using EB-R (L) and cEB-R (R)

Figure 7.4: The saliency maps produced by EB-R (left) and cEB-R (right) together with the *THUMOS14* groundtruth bounding box over the same frame of the actions (a) *Surfing* and (b) *BasketballDunk*. In both cases, EB-R highlights the most salient regions of the frame for this action (human), which matches the bounding box annotation. However, cEB-R highlights the region that is unique to the ground truth action: the waves for *Surfing*, and the hoop for *BasketballDunk*. This is because highlighting the human region does not provide insightful information to the classifier.

combination with a different random background video. The *THUMOS14* dataset consists of 1010 untrimmed validation videos and 1574 untrimmed test videos of 20 action classes. Among test videos, we evaluate our framework on the 213 test videos which contain annotations as in [137, 104].

Baselines. For the synthetic experiment, we compare cEB-R and EB-R with a probability-based approach where we threshold the predicted probability (to 1 if ≥ 0.5, to -1 if < 0.5) of the *GT* class at every time-step. For the detection experiment in *THUMOS14* we compare our proposed method with state-of-the-art approaches.

Models. For the synthetic dataset, we use the same CNN-LSTM model used for spatial action grounding (Sec. 7.4.1). For the *THUMOS14* dataset we use a CNN-LSTM model: the same VGG-16 model used for spatial action grounding (Sec. 7.4.1) fine-tuned with a



(a) Grounding of the action *TableTennisShot* in the video

(b) Grounding of the action *Skiing* in the video

Figure 7.5: Applying *contrastive* Excitation Backprop for Recurrent Networks (cEB-R) to produce spatiotemporal localization of actions in sample frames of a video. Demonstrated here is (a) cEB-R spatiotemporal localization of *TableTennisShot* in a video (b) cEB-R spatiotemporal localization of *Skiing* in the same video. The video consists of two consecutive actions that are synthetically concatenated: *Skiing* followed by *TableTennisShot*.

one-layer LSTM on *UCF101* and trimmed sequences from *THUMOS14* background and validation sets.

Setup and Results: Synthetic Data. First, we perform experiments on the synthetic videos composed of two sequential actions, where the boundary is the midpoint. Figure 7.5 presents a sample spatiotemporal localization. The heatmaps produced by cEB-R correctly ground the queried action. While Figure 7.5 presents a qualitative sample, Figure 7.6 quantitatively presents results on the entire test set. The action switches from *GT* to *rand* or vice versa midway. It can be seen that the sum of saliency maps is: positive and increasing as more of the *GT* action is observed, negative and decreasing as more of the *rand* action is observed.

Next, we perform experiments where we vary the length of the *GT* action that we want to localize inside a clip. To retain action dynamics, we sample *GT* and *rand* from the entire length of their corresponding videos. Table 7.2 presents the temporal localization

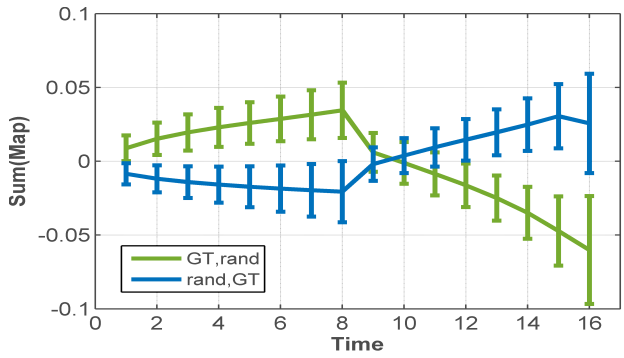


Figure 7.6: Sum of the saliency maps at *fc7* over time, in frames, for synthetic videos that (blue) have a *rand* action followed by a *GT* action and (green) have a *GT* followed by a *rand* action. The average and standard deviation are reported over all test videos. cEB-R provides an accurate midway boundary between actions.

results of our synthetic data. In the experimental setup with fixed action length we assume that we know the label and length of the action to be localized. To localize, we find the highest consecutive sum of attention maps for the desired action length. Regarding the sequences with unknown action lengths, we only assume the label of the action to be localized and perform the pipeline described in Sec. 7.2. In the bottom half of Table 7.2 we only report thresholded probabilities and cEB-R results since our localization procedure assumes negative values at action boundaries, whereas EB-R is non-negative. The grounded evidence obtained by cEB-R attains the highest detection scores, 73.5% and 62.0%, for action sequences of known and unknown lengths, respectively, for IoU overlap between detections and ground-truth of $\alpha = 0.5$, despite the fact that the model is not trained for localization.

Setup and Results: *THUMOS14* Pointing Game. We evaluate the pointing game in time for *THUMOS14* -a fair evaluation for methods that do not optimize for detection. For processing, we divide a video into 128-frame consecutive clips. We perform the pointing game by pointing [148] in time to the peak sum of saliency maps. For each ground-truth annotation we check if the detected peak is within its boundaries. If yes, we count it as a

		Length (frames)	Probability (%)	EB-R (%)	cEB-R (%)
Action Length	***Known***	11	8.5	11.3	**15.5**
		41	28.2	38.5	**53.2**
		65	47.7	56.3	**73.5**
	Unknown	11	3.4	-	**4.1**
		41	9.5	-	**47.9**
		65	35.7	-	**62.0**

Table 7.2: Action detection results on synthetic data, measured by mAP at IoU threshold $\alpha = 0.5$. Top part of table: methods assume that the length and label of the action to be detected are known. Bottom part of table: methods assume that the label is known, but the length is unknown. cEB-R attains best performance.

Method	Accuracy (%)
Random	57.3
Peak probability	65.8
cEB-R	65.1
Peak probability + cEB-R	**77.4**

Table 7.3: Pointing game in time performed on the *THUMOS14* test set. The probability of an action together with the evidence for presence of the action are complementary and give a great improvement in accuracy when combined.

hit, otherwise, as a miss. We compare this approach with the peak position of predicted probabilities, and a random point in that clip.

The results of this experiment are presented in Table 7.3. Pointing to a random position clearly obtains lowest results while peak probability (65.8%) and cEB-R (65.1%) have similar performance. However, peak probability does not offer spatial localization. Peak probability uses the model prediction, while cEB-R uses the evidence of that prediction. Moreover, we observe that peak probability and cEB-R are complementary, yielding 77.4%.

Setup and Results: *THUMOS14* Action Detection. We evaluate how well our grounding does on the more challenging task of action detection that it was not trained for. In this experiment, we divide a video into 128-frame consecutive clips for processing. Table 7.4

Method	mAP ($\alpha = 0.1$)
Karaman *et al.* [53]	4.6
Wang *et al.* [126]	18.2
Oneata *et al.* [85]	36.6
Richard *et al.* [102]	39.7
Shou *et al.* [105]	47.7
Yeung *et al.* [143]	48.9
Yuan *et al.* [144]	51.4
Xu *et al.* [137]	54.5
Zhao *et al.* [153]	60.3
Kaufman *et al.* [56]	61.1
Ours	57.9

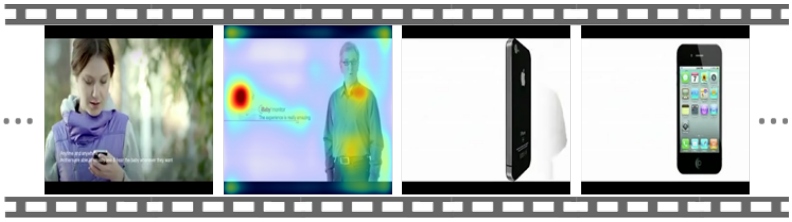
Table 7.4: Our approach *vs.* fully supervised approaches for action detection on *THUMOS14*, measured by mAP at IoU threshold $\alpha = 0.1$. Although our model is not trained for action detection (trained for recognition), we achieve 57.9%, which is comparable to state-of-the-art when localizing a ground truth action in a video.

presents the temporal detection results of the *THUMOS14* dataset. Differently from the pointing game experiment, we detect the start and end of the ground-truth action. We note that although our method is not supervised for the detection task, we achieve an accuracy of 57.9% when locating a ground truth class with an overlap $\alpha = 0.1$ as demonstrated in Table 7.4.

7.5 Experiments: Caption Grounding

In this section, we show how cEB-R is also applicable in the context of caption grounding. As observed by [97], there is an absence of datasets with spatiotemporal annotations of frames for captions. Therefore, they propose the following experimental setup which we follow: qualitative results for the spatiotemporal grounding on videos, and quantitative results for spatial grounding on images.

Datasets. We use the *MSR-VTT* [138] dataset for video captioning and *Flickr30kEntities* [96] for image captioning.



(a) grounding of the word *man*

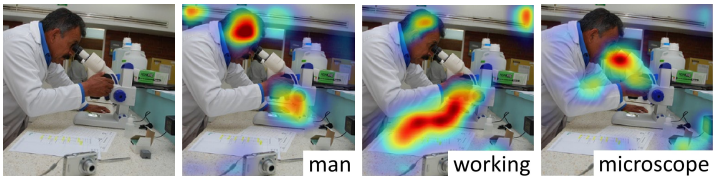
(b) grounding of the word *phone*

Figure 7.7: Comparison of grounding of words *man* and *phone* in the caption *A man is talking about a phone* of a video from *MSR-VTT* using cEB-R. The man is well localized in (a) and the phone is well localized in (b), as desired.

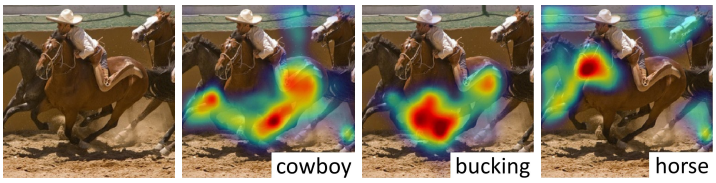
Models. We use the CNN-LSTM-LSTM video captioning model of [119] trained on *MSR-VTT* to test our cEB-R approach for spatiotemporal grounding as described in Sec. 7.3. We use the same video captioning model, without the average pooling layer, trained on *Flickr30kEntities* for image captioning. The models have comparable METEOR scores to the Caption-Guided Saliency work of [97], to which we compare our results: 26.5 (*vs.* 25.9) for video captioning and 18.0 (*vs.* 18.3) for image captioning.

Setup and Results. For the *MSR-VTT* video dataset, we sample 26 frames per video following [97] and perform grounding of nouns. Figure 7.7 presents the grounding for the word *man* and *phone* in the same video. The *man* is well localized only in frames where a man appears, and the *phone* is well localized in frames where a phone appears.

We quantitatively evaluate our results of spatial grounding using the pointing game on the *Flickr30kEntities* and compare our method to the Caption-Guided Saliency work of [97], following their evaluation protocol. We use ground truth captions as an input to our model in order to reproduce the same captions. Then, we use bounding box annotations for



(a) image caption: *A man in a lab coat is working on a microscope.*



(b) image caption: *A cowboy is riding a bucking horse.*

Figure 7.8: Grounding different words of a caption using cEB-R for two images from the *Flicker30kEntities* dataset.

Method	Avg (Noun Phrases)
Baseline random	0.268
Baseline center	0.492
Caption-Guided Saliency [97]	0.501
Ours	**0.512**

Table 7.5: Evaluation of spatial saliency on *Flickr30kEntities* using cEB-R. Baseline random samples the maximum point uniformly and Baseline center always picks the center.

each noun phrase in the ground truth captions and check whether the maximum point in a saliency map is inside the annotated bounding box.

Table 7.5 shows the results of the spatial pointing game on *Flickr30kEntities*. Our approach achieves comparable performance to [97]. In this experiment, we ground the ground truth captions to match the experimental setup in [97]. Although we follow their protocol for fair comparison, we note that our method can better highlight evidence using generated captions (*vs.* ground truth captions). This is because the evidence of a ground truth noun that is not predicted may not be sufficiently activated in the forward pass. Figure 7.8 presents some visual examples of grounding in images using the generated captions.

Our approach has a computational advantage over [97]. In order to obtain spatial

saliency maps for a word in a video, c-EB-R requires one forward pass and one backward pass through the CNN-LSTM-LSTM, while [97] requires one forward pass through the CNN part, but m forward passes through the LSTM-LSTM part, where $m = 64$ is the area of the saliency map (*vs.* our single backward pass). Moreover, they require f forward LSTM passes, where $f = 26$ is the number of frames, to compute the temporal grounding, whereas ours is implicitly spatiotemporal.

7.6 Application: Reflecting the Abstraction Capability of Models

We recently introduced the Moments in Time Dataset [82], a large-scale human-annotated collection of one million short videos corresponding to dynamic events unfolding within three seconds. Temporal events of such length correspond to the average duration of human working memory [6, 9], specialized in representing information that is changing over time.

Modeling the dynamics even for actions occurring in short videos poses many challenges including the fact that meaningful events do not include only people, but also objects, animals, and natural phenomena. This dataset represents a dynamical event at different levels of abstraction. For instance, inspecting videos in the dataset labeled with the action "opening", one can find people opening doors, gates, drawers, curtains and presents, animals and humans opening eyes, mouths and arms, and even a flower opening its petals. This is illustrated in Figure 7.9. This first version of the Moments in Time dataset includes one action label per video, and 339 different action classes. The classes are chosen such that they include the most commonly used verbs in the English language, covering a wide and diverse semantic space.

The challenge is to develop models that recognize transformations in a way that will allow them to discriminate between different actions, yet generalize to other actors performing or undergoing the same action under different settings. Typically, classification accuracy is

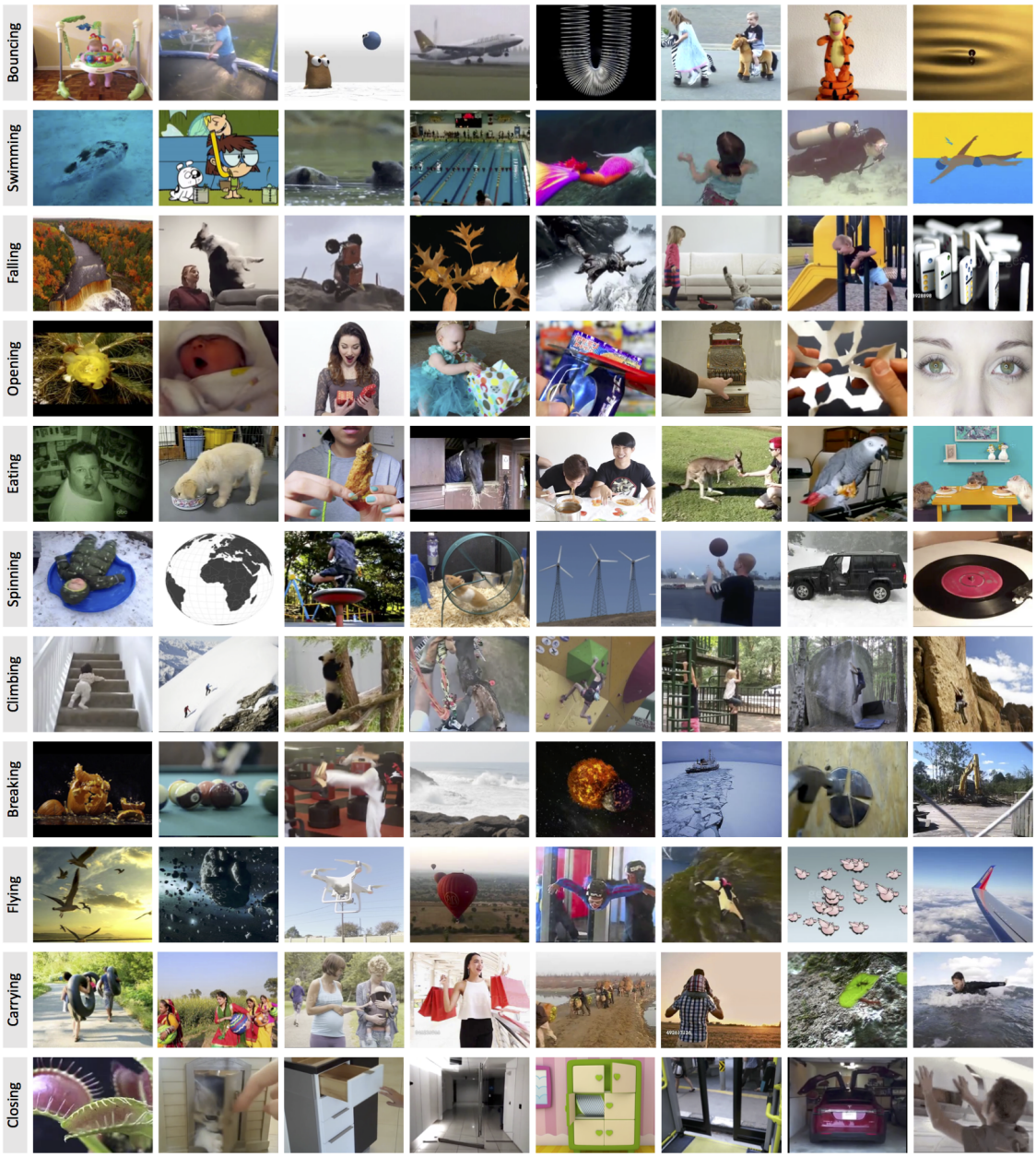


Figure 7.9: Day-to-day events can happen to many types of actors, in different environments, and at different scales. Moments in Time dataset [82] has a significant intra-class variation among the categories. Here we illustrate one frame for a few video samples and actions. For example, car engines can open, books can open, and tulips can open.

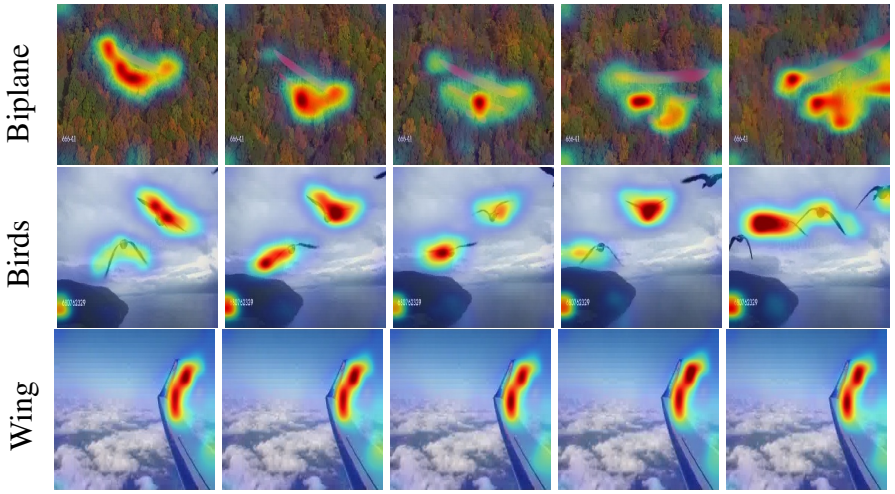


Figure 7.10: Sample grounding results for three test videos of the class *flying* from the Moments in Time Dataset to visualize the cues a CNN-LSTM model uses for classification.

reported to summarize the recognition capability of models on classification datasets. However, classification accuracy alone, unless 100%, is not indicative as to whether the models are really modeling this diversity of actors. A classifier may be incorrectly classifying a whole subset of cases/actors.

For example, it may be that all the correctly classified instances of the class *bouncing* are bouncing balls while all other bouncing actors are misclassified. A ballerina may be spinning, and a toy may be spinning; the question becomes, does the model in both cases "look" at the spinning object? or does it only correctly focus its attention on the person/object spinning based on frequencies of occurrence in the dataset?

We now demonstrate how grounding can be used for visually analyzing a model's performance on such datasets. We train a temporal CNN-LSTM model for the classification task of the Moments in Time Dataset. We then perform cEB-R to highlight the evidence the model used to make its prediction. We present sample results for the action *flying* in Figure 7.10. We present video frames of three videos, a biplane flying, birds flying, and flying inside an airplane. In all three cases, the cues the model is using to discriminate

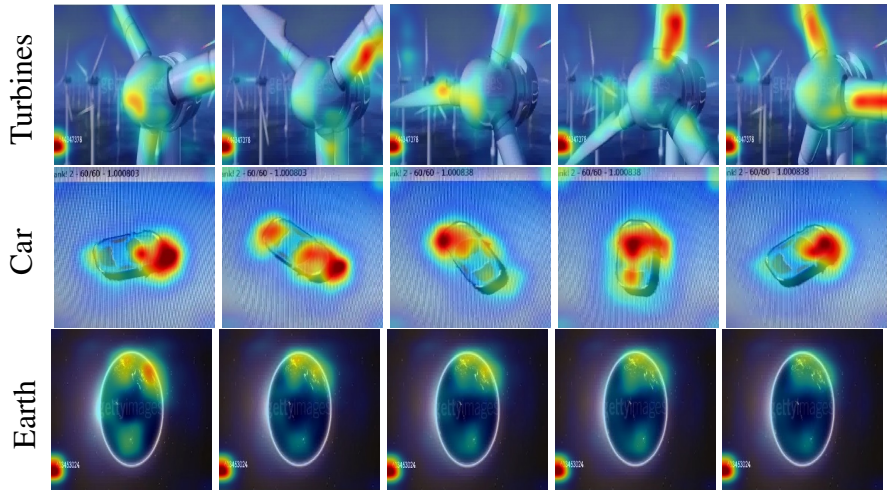


Figure 7.11: Sample grounding results for three test videos of the class *spinning* from the Moments in Time Dataset to visualize the cues a CNN-LSTM model uses for classification.

flying are depicted. This suggests that the model is able to understand that birds can fly and airplanes can fly, or that the action flying can be performed by multiple actors. We present sample results from the classes *spinning* and *opening* in Figures 7.11 and 7.12, respectively. We observe that the trained temporal model is capable of recognizing the action spinning for several actors: turbines, car, earth. Similarly, the model is able to recognize the action opening for several actors: flower, box, can.

7.7 Discussion

In this chapter, we formulate top-down attention for recurrent neural network models for spatiotemporal grounding. We do so using a single contrastive backpropagation pass at test time. We demonstrate that spatiotemporal grounding performed on models trained for action classification can be utilized to perform coarse action localization without being trained to do so. We also demonstrate spatiotemporal grounding results for the image captioning and video captioning tasks. In addition, we demonstrate how spatiotemporal grounding can be a useful visualization tool for reflecting the abstraction capability of

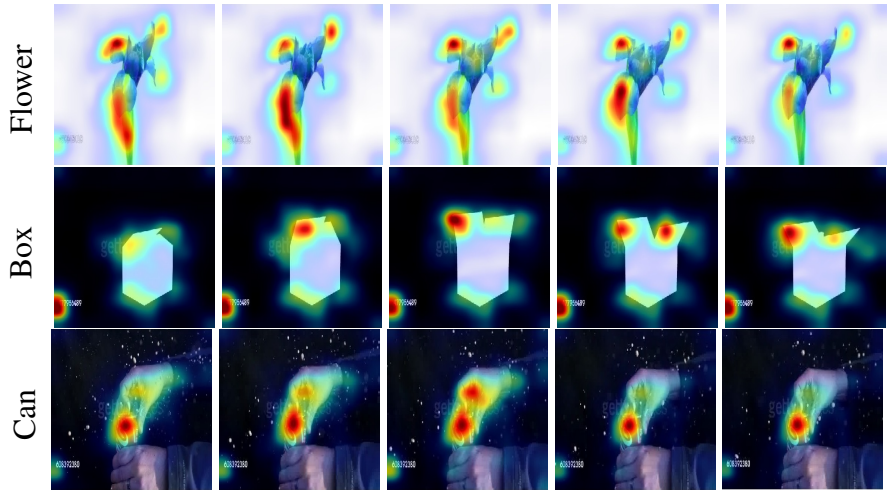


Figure 7.12: Sample grounding results for three test videos of the class *opening* from the Moments in Time Dataset to visualize the cues a CNN-LSTM model uses for classification.

models that are trained on datasets possessing labels at a higher level of abstraction.

Chapter 8

Conclusions and Future Work

In this chapter we present the main contributions of this thesis: improving video classification using the image modality without grounding; proposing a novel dropout regularizer for deep models based on spatial grounding; proposing a novel evidence evaluation scheme for improving fine-grained classification; proposing the first top-down saliency formulation for spatiotemporal grounding. We then discuss the strengths and limitations of the proposed approaches, and present interesting directions for future work.

8.1 Main Contributions

We start by studying the benefits of web images that are in one-to-one action category correspondence with training videos. We collect three datasets of web action images: the BU101 *filtered* dataset and two *unfiltered* datasets, BU101-unfiltered and BU203-unfiltered. We show that using web action images in training CNN models for action recognition is an effective and low-cost approach to improve performance. While videos contain a lot of useful temporal information to describe an action, and while it is more beneficial to use videos only than to use web images only, web images can provide *complementary* information to a finite set of videos allowing for a significant reduction in the video data required for training. We observe that this complementarity is insensitive to different CNN architectures and is evident in many kinds of actions. Both filtered and unfiltered web

action images are complementary to video training data. However, human filtering of the web action images is still useful: considerably fewer filtered images are required to achieve similar performance improvements. Using web action images can also boost the efficiency of CNN training. When using the same number of training samples, the trained model can achieve significantly higher recognition performance if half of the samples are web images. We also show that, to achieve the same recognition performance, we can greatly reduce the number of training videos and use unfiltered web action images instead. For CNN finetuning, using web action images as training data in addition to training video frames can greatly reduce the number of conservative filters, *i.e.* CNN filters that undergo minimal changes and have low activation on video frames. Such conservative filters reduce the effective number of parameters in the CNN model and thus may be harmful for its modeling capacity. We speculate that one underlying mechanism that delivers the benefits of web action images, as shown in our experiments, is that reducing the number of conservative filters enables *re-use* of those parameters for modeling visual patterns of the new task, which in this work is action recognition.

We found some interesting misclassifications that led us to explore saliency-based techniques to explain such misclassifications. We presented several applications of saliency-based techniques in explaining correct/incorrect model predictions. We demonstrated how such techniques can highlight differences between domains, and shift of model focus before and after domain adaptation. This led to the proposal of two novel techniques that both lead to improved generalization of deep convolutional neural networks: a regularization scheme (Chapter 5), and an evidence evaluation scheme (Chapter 6).

We propose a new regularization scheme, Excitation Dropout, that encourages the learning of alternative paths (re-wiring) in a neural network by deliberately paralyzing high-saliency neurons that contribute more to a network's prediction during training. High-

saliency is determined by spatial grounding of the network during the training procedure. In extensive experiments on four image/video recognition datasets, and on different architectures, we demonstrate that Excitation Dropout yields better generalization on unseen data. Our approach consistently results in an improved utilization of the network neurons reflected by different metrics. Our approach also demonstrates higher robustness as more neurons are switched off during a network compression procedure. We also demonstrate this visually through the ability to recover more of the saliency map even when a high percentage of the most salient neurons is dropped-out. This ability further reflects the alternative learnt paths.

We devise a methodology that utilizes explicit spatial grounding to refine a model's prediction at test time. Our refinement module selects one of the top-k model predictions based on which has the most reasonable (evidence, prediction) pair; defined as the most consistent with respect to a pre-defined pool generated once using adversarial erasing of a grounding technique (`Guided Zoom`), and another using an ensemble of grounding techniques (`Ensemble Guided Zoom`). We find that both pool generation techniques improve a base model's prediction accuracy similarly, and therefore demonstrate analogous complementarity of localized salient regions.

We devise a temporal formulation that enables us to visualize how recurrent networks ground their decisions in visual content. Our formulation employs a single backward pass to produce saliency maps that highlight the evidence that a given recurrent model uses in generating its output predictions. We apply this to two video understanding tasks: video action recognition, and video captioning. We demonstrate how spatiotemporal top-down saliency is capable of grounding evidence on several action and captioning datasets. These datasets provide annotations for detection and/or localization, to which we have compared the evidence in our generated saliency maps that are generated without explicit training

for such detection/localization tasks. We observe the strengths of cEB-R in highlighting discriminative evidence, which was particularly beneficial for temporal grounding. We also observe the strengths of its variant, EB-R, in highlighting salient evidence, which was particularly beneficial for spatial localization of action subjects. We also present how our approach can be used for reflecting the abstraction capability of models that target classifying classes that are highly abstract.

8.2 Limitations and Interesting Directions for Future Work

We start in Chapter 3 by studying how information that is available in the image domain can be used to gain improved accuracy in the classification of actions in the video domain. The main limitation of this work, is that the resulting classification results could not be directly grounded. We then introduce spatial grounding techniques to highlight the regions in the input image or video frame responsible for such classifications (Chapter 4).

8.2.1 Spatial Grounding

In Chapters 5 and 6 we introduce two frameworks that utilize spatial grounding to improve classification of deep models during training and once training is complete. The main strengths of our frameworks are that they are general and can be applied to any network. However, the main limitation is that we explicitly compute saliency maps to ground predictions using the internal representation of a trained model. Future work includes learning to ground without explicitly computing saliency maps. It would be interesting to explore whether we can train vision models that make a prediction and provide a visual and/or textual explanation for it.

One open question is: *How do we evaluate the accuracy of grounding?* Currently, most grounding techniques are evaluated against bounding box annotations of where the action

or object class occur in the image or video frame. However, this may not be the best way for such evaluation. We give an example to demonstrate why: Grounding the action *Surfing* using two variants of a grounding algorithm (demonstrated in Figure 7.4) highlight different regions. In one case the man performing the surfing action is highlighted, and in the other case the waves are highlighted. The method highlighting the waves will be penalized for its grounding result. However the waves are a very discriminative feature of the action *Surfing*, particularly if the dataset does not contain other actions that involve high waves. Therefore, an interesting research direction would be exploring what constitutes a correct grounding and how a grounding technique can be evaluated. Some initial efforts in this direction are underway [3].

In addition, the current literature lacks sufficient study of how models of different architectures affect the grounding result using a specific grounding technique, and how much different grounding techniques agree on the evidence upon which the same model makes decisions. It is also interesting to explore to what extent models of the same architecture that are trained with different initializations, and converge to different local minima, agree on their grounding. Referencing the surfing example, it may be reasonable for one local minima to use the waves as a discriminative evidence for the action *Surfing* and for another to use the person performing the action as the evidence.

In Chapter 4, we demonstrated that grounding techniques are capable of highlighting the differences between domains and that grounding techniques can capture how the focus of a model changes using different training strategies: without domain adaptation, and with domain adaptation. This serves as a feasibility study for a natural extension of our theme 'Grounding for better classification' to bridge the gap between different domains.

8.2.2 Spatiotemporal Grounding

In Chapter 7 we extend grounding to become spatiotemporal in recurrent neural networks for videos. The main strength of this work is that the resulting grounding can be used to perform coarse object/action localization, while the model was only trained on the recognition/classification task of trimmed video sequences. Another strength of this work is that the resulting grounding is obtained using a single backward pass through the recurrent model. A natural extension of this work is to extend the theme of 'Grounding for improved classification' that we introduced for the spatial domain to the spatiotemporal domain, in which spatiotemporal grounding would be utilized to improve video classification and captioning tasks in deep models.

Other future work related to spatiotemporal grounding includes a study of how grounding will be affected/guided by natural language descriptions of action activities in videos, particularly fine-grained action activities such as those present in the Something-Something dataset [77]. We believe that guiding visual attention in images and videos using textual descriptions might enable models to reason about visual inputs in a manner that is similar to what humans do, leading to models that are guided to focus on specific image/video regions regardless of their architecture and regardless of which local minima they converge to.

While we worked with different types of outputs, we focused on visual data (images and videos) for the input modality. Future work would also include using different modalities as input such as text, audio, time-series, or geospatial data. For example, a sample application for textual input would be spatiotemporally highlighting word alignment for the task of machine translation, as attention has demonstrated to not be a very accurate measure of such alignment [61].

Analogous to the case of spatial grounding, judging the correctness of a spatiotemporal grounding remains an open problem. Let us consider the action '*pouring water from a*

cup'. It is unclear whether the correct grounding should highlight the hand performing the action, the cup undergoing the action, the water being poured from the cup, or all of them together. It would also be interesting to explore how the grounding changes as a function of the description to focus on the different actors involved in the action.

Bibliography

[1] ABU-EL-HAIJA, S., KOTHARI, N., LEE, J., NATSEV, P., TODERICI, G., VARADARAJAN, B., AND VIJAYANARASIMHAN, S. Youtube-8M: A large-scale video classification benchmark. *arXiv preprint arXiv:1609.08675* (2016). 2

[2] ACHILLE, A., AND SOATTO, S. Information dropout: Learning optimal representations through noisy computation. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* (2018). 19

[3] ADEBAYO, J., GILMER, J., MUELLY, M., GOODFELLOW, I., HARDT, M., AND KIM, B. Sanity checks for saliency maps. In *Advances in Neural Information Processing Systems (NIPS)* (2018), pp. 9524–9535. 126

[4] BA, J., AND FREY, B. Adaptive dropout for training deep neural networks. In *Advances in Neural Information Processing Systems (NIPS)* (2013). 20, 68, 72

[5] BACH, S., BINDER, A., MONTAVON, G., KLAUSCHEN, F., MÜLLER, K.-R., AND SAMEK, W. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS One 10*, 7 (2015), e0130140. 17, 18

[6] BADDELEY, A. Working memory. *Science:255.556-559* (1992). 117

[7] BALUCH, F., AND ITTI, L. Mechanisms of top-down attention. *Trends in Neurosciences 34*, 4 (2011), 210–224. 17

[8] BARGAL, S. A., ZUNINO, A., KIM, D., ZHANG, J., MURINO, V., AND SCLAROFF, S. Excitation backprop for RNNs. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (June 2018). 7

[9] BARROUILLET, P., BERNARDIN, S., AND CAMOS, V. Time constraints and resource sharing in adults' working memory spans. *Journal of Experimental Psychology: General:133.83* (2004). 117

[10] BARSOUM, E., ZHANG, C., CANTON FERRER, C., AND ZHANG, Z. Training deep networks for facial expression recognition with crowd-sourced label distribution. In *Proc. ACM International Conference on Multimodal Interaction (ICMI)* (2016). 5, 53

[11] BAZZANI, L., BERGAMO, A., ANGUELOV, D., AND TORRESANI, L. Self-taught object localization with deep networks. In *Applications of Computer Vision (WACV), 2016 IEEE Winter Conference on* (2016), IEEE, pp. 1–9. 18

[12] BECK, D. M., AND KASTNER, S. Top-down and bottom-up mechanisms in biasing competition in the human brain. *Vision Research 49*, 10 (2009), 1154–1165. 17

[13] BRANSON, S., VAN HORN, G., BELONGIE, S., AND PERONA, P. Bird species categorization using pose normalized deep convolutional nets. *arXiv preprint arXiv:1406.2952* (2014). 20, 95

[14] CABA HEILBRON, F., ESCORCIA, V., GHANEM, B., AND CARLOS NIEBLES, J. Activitynet: A large-scale video benchmark for human activity understanding. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2015), pp. 961–970. 27, 37, 38, 45, 50

[15] CAO, C., LIU, X., YANG, Y., YU, Y., WANG, J., WANG, Z., HUANG, Y., WANG, L., HUANG, C., XU, W., ET AL. Look and think twice: Capturing top-down visual attention with feedback convolutional neural networks. In *Proc. IEEE International Conference on Computer Vision (ICCV)* (2015), pp. 2956–2964. 4, 7, 17, 18, 99

[16] CHATFIELD, K., SIMONYAN, K., VEDALDI, A., AND ZISSERMAN, A. Return of the devil in the details: Delving deep into convolutional nets. In *Proc. British Machine Vision Conference (BMVC)* (2014). 25, 33, 34, 45

[17] CHEN, C.-Y., AND GRAUMAN, K. Watching unlabeled video helps learn new human actions from very few labeled snapshots. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2013). 16

[18] CHEN, D., HUA, G., WEN, F., AND SUN, J. Supervised transformer network for efficient face detection. *arXiv preprint arXiv:1607.05477* (2016). 52

[19] CUI, Y., ZHOU, F., WANG, J., LIU, X., LIN, Y., AND BELONGIE, S. Kernel pooling for convolutional neural networks. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2017). 21, 94

[20] DENG, J., DONG, W., SOCHER, R., LI, L.-J., LI, K., AND FEI-FEI, L. Imagenet: A large-scale hierarchical image database. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2009). 32, 70

[21] DENG, J., KRAUSE, J., AND FEI-FEI, L. Fine-grained crowdsourcing for fine-grained recognition. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2013), pp. 580–587. 21, 83

[22] DESIMONE, R. Visual attention mediated by biased competition in extrastriate visual cortex. *Philosophical Transactions of the Royal Society of London B: Biological Sciences 353*, 1373 (1998), 1245–1255. 17

[23] DESIMONE, R., AND DUNCAN, J. Neural mechanisms of selective visual attention. *Annual Review of Neuroscience 18*, 1 (1995), 193–222. 17

[24] DHALL, A., GOECKE, R., JOSHI, J., HOEY, J., AND GEDEON, T. EmotiW 2016: Video and group-level emotion recognition challenges. In *Proc. ACM International Conference on Multimodal Interaction (ICMI)* (2016). 54

[25] DHALL, A., GOECKE, R., LUCEY, S., AND GEDEON, T. Collecting large, richly annotated facial-expression databases from movies. *IEEE MultiMedia 19*, 3 (2012), 34–41. 51, 58

[26] DONAHUE, J., ANNE HENDRICKS, L., GUADARRAMA, S., ROHRBACH, M., VENUGOPALAN, S., SAENKO, K., AND DARRELL, T. Long-term recurrent convolutional networks for visual recognition and description. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2015), pp. 2625–2634. 105

[27] DUAN, L., XU, D., AND CHANG, S.-F. Exploiting web images for event recognition in consumer videos: A multiple source domain adaptation approach. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2012). 16

[28] EVERINGHAM, M., VAN GOOL, L., WILLIAMS, C. K., WINN, J., AND ZISSERMAN, A. The pascal visual object classes (VOC) challenge. *International Journal of Computer Vision (IJCV) 88*, 2 (2010), 303–338. 30

[29] FANG, H., GUPTA, S., IANDOLA, F., SRIVASTAVA, R. K., DENG, L., DOLLÁR, P., GAO, J., HE, X., MITCHELL, M., PLATT, J. C., ET AL. From captions to visual concepts and back. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2015). 18

[30] FONG, R. C., AND VEDALDI, A. Interpretable explanations of black boxes by meaningful perturbation. In *Proc. IEEE International Conference on Computer Vision (ICCV)* (2018). 8, 17, 22

[31] FU, J., ZHENG, H., AND MEI, T. Look closer to see better: Recurrent attention convolutional neural network for fine-grained image recognition. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2017). 21, 83, 84, 89, 94, 95, 96, 97

[32] GIRSHICK, R., DONAHUE, J., DARRELL, T., AND MALIK, J. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2014). 39

[33] GREFF, K., SRIVASTAVA, R. K., KOUTNÍK, J., STEUNEBRINK, B. R., AND SCHMIDHUBER, J. LSTM: A search space odyssey. *IEEE transactions on neural networks and learning systems* (2016). 102

[34] GRIFFIN, G., HOLUB, A., AND PERONA, P. Caltech-256 object category dataset. Tech. Rep. 7694, California Institute of Technology, 2007. 69

[35] GUILLAUMIN, M., KÜTTEL, D., AND FERRARI, V. Imagenet auto-annotation with segmentation propagation. *International Journal of Computer Vision (IJCV) 110*, 3 (2014), 328–348. 18

[36] GUPTA, A., KEMBHAVI, A., AND DAVIS, L. S. Observing human-object interactions: Using spatial and functional compatibility for recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI) 31*, 10 (2009), 1775–1789. 30

[37] HE, K., ZHANG, X., REN, S., AND SUN, J. Deep residual learning for image recognition. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016), pp. 770–778. 53, 94

[38] HEBB, D. O. *The organization of behavior: A neuropsychological theory*. Psychology Press, 2005. 6, 65

[39] HINTON, G., VINYALS, O., AND DEAN, J. Distilling the knowledge in a neural network. In *NIPS Deep Learning and Representation Learning Workshop* (2015). 67

[40] HINTON, G. E., SRIVASTAVA, N., KRIZHEVSKY, A., SUTSKEVER, I., AND SALAKHUTDINOV, R. Improving neural networks by preventing co-adaptation of feature detectors. *CoRR abs/1207.0580* (2012). 6, 19, 65, 67

[41] HUANG, S., XU, Z., TAO, D., AND ZHANG, Y. Part-stacked CNN for fine-grained visual categorization. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016). 20, 83

[42] HUANG, W., BRIDGE, C. P., NOBLE, J. A., AND ZISSERMAN, A. Temporal HeartNet: Towards human-level automatic analysis of fetal cardiac screening video. *arXiv preprint arXiv:1707.00665* (2017). 60

[43] IKIZLER-CINBIS, N., CINBIS, R. G., AND SCLAROFF, S. Learning actions from the web. In *Proc. IEEE International Conference on Computer Vision (ICCV)* (2009). 30

[44] IKIZLER-CINBIS, N., AND SCLAROFF, S. Web-based classifiers for human action recognition. *IEEE Transactions on Multimedia 14*, 4 (2012), 1031–1045. 16

[45] JADERBERG, M., SIMONYAN, K., ZISSERMAN, A., ET AL. Spatial transformer networks. In *Advances in Neural Information Processing Systems (NIPS)* (2015). 21, 83, 84

[46] JAMALUDIN, A., KADIR, T., AND ZISSERMAN, A. SpineNet: Automated classification and evidence visualization in spinal MRIs. *Medical Image Analysis 41* (2017), 63–73. 60

[47] JI, S., XU, W., YANG, M., AND YU, K. 3D convolutional neural networks for human action recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI) 35*, 1 (2013), 221–231. 16

[48] JIA, Y., SHELHAMER, E., DONAHUE, J., KARAYEV, S., LONG, J., GIRSHICK, R., GUADARRAMA, S., AND DARRELL, T. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093* (2014). 45, 47

[49] JIANG, Y.-G., LIU, J., ROSHAN ZAMIR, A., LAPTEV, I., PICCARDI, M., SHAH, M., AND SUKTHANKAR, R. THUMOS challenge: Action recognition with a large number of classes. http://crcv.ucf.edu/ICCV13-Action-Workshop/, 2013. 45

[50] JIANG, Y.-G., LIU, J., ROSHAN ZAMIR, A., TODERICI, G., LAPTEV, I., SHAH, M., AND SUKTHANKAR, R. THUMOS challenge: Action recognition with a large number of classes. http://crcv.ucf.edu/THUMOS14/, 2014. 108, 109

[51] JOZEFOWICZ, R., ZAREMBA, W., AND SUTSKEVER, I. An empirical exploration of recurrent network architectures. In *Proc. International Conference on Machine Learning (ICML)* (2015). 102

[52] KANG, G., LI, J., AND TAO, D. Shakeout: A new approach to regularized deep neural network training. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)* (2017). 19

[53] KARAMAN, S., SEIDENARI, L., AND DEL BIMBO, A. Fast saliency based pooling of Fisher encoded dense trajectories. In *ECCV THUMOS Workshop* (2014), vol. 1, p. 5. 114

[54] KARPATHY, A., JOHNSON, J., AND FEI-FEI, L. Visualizing and understanding recurrent networks. In *Proc. International Conference on Learning Representations (ICLRw)* (2016). 8, 23, 99

[55] KARPATHY, A., TODERICI, G., SHETTY, S., LEUNG, T., SUKTHANKAR, R., AND FEI-FEI, L. Large-scale video classification with convolutional neural networks. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2014). 2, 16, 25, 32, 48, 49

[56] KAUFMAN, D., LEVI, G., HASSNER, T., AND WOLF, L. Temporal tessellation: A unified approach for video analysis. In *Proc. IEEE International Conference on Computer Vision (ICCV)* (2017). 114

[57] KHOSLA, A., JAYADEVAPRAKASH, N., YAO, B., AND LI, F.-F. Novel dataset for fine-grained image categorization: Stanford dogs. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition Workshop (CVPRw)* (2011). 83, 94

[58] KHURRAM SOOMRO, A. R. Z., AND SHAH, M. UCF101: A dataset of 101 human action classes from videos in the wild. In *CRCV-TR-12-01* (2012). 27

[59] KINGMA, D. P., SALIMANS, T., AND WELLING, M. Variational dropout and the local reparameterization trick. In *Advances in Neural Information Processing Systems (NIPS)* (2015). 19

[60] KOCH, C., AND ULLMAN, S. Shifts in selective visual attention: Towards the underlying neural circuitry. In *Matters of Intelligence*. Springer, 1987, pp. 115–141. 17

[61] KOEHN, P., AND KNOWLES, R. Six challenges for neural machine translation. *arXiv preprint arXiv:1706.03872* (2017). 127

[62] KRAUSE, J., JIN, H., YANG, J., AND FEI-FEI, L. Fine-grained recognition without part annotations. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2015). 20, 95

[63] KRAUSE, J., SAPP, B., HOWARD, A., ZHOU, H., TOSHEV, A., DUERIG, T., PHILBIN, J., AND FEI-FEI, L. The unreasonable effectiveness of noisy data for fine-grained recognition. In *Proc. European Conference on Computer Vision (ECCV)* (2016). 21, 84, 94

[64] KRAUSE, J., STARK, M., DENG, J., AND FEI-FEI, L. 3D object representations for fine-grained categorization. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition Workshop (CVPRw)* (2013). 83

[65] KRIZHEVSKY, A. Learning multiple layers of features from tiny images. Tech. rep., 2009. 69

[66] KRIZHEVSKY, A., SUTSKEVER, I., AND HINTON, G. E. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems (NIPS)* (2012), pp. 1097–1105. 25, 34, 43, 46, 70, 73

[67] LAN, Z., LIN, M., LI, X., HAUPTMANN, A. G., AND RAJ, B. Beyond gaussian pyramid: Multi-skip feature stacking for action recognition. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2015), pp. 204–212. 16, 49

[68] LAPTEV, I., MARSZAŁEK, M., SCHMID, C., AND ROZENFELD, B. Learning realistic human actions from movies. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2008). 15

[69] LEVI, G., AND HASSNER, T. Emotion recognition in the wild via convolutional neural networks and mapped binary patterns. In *Proc. ACM International Conference on Multimodal Interaction (ICMI)* (2015), ACM, pp. 503–510. 58

[70] LI, J., CHEN, X., HOVY, E., AND JURAFSKY, D. Visualizing and understanding neural models in nlp. In *NAACL-HLT* (2016), pp. 681–691. 8, 23

[71] LI, K., WU, Z., PENG, K.-C., ERNST, J., AND FU, Y. Tell me where to look: Guided attention inference network. *arXiv preprint arXiv:1802.10171* (2018). 7

[72] LI, Z., GONG, B., AND YANG, T. Improved dropout for shallow and deep learning. In *Advances in Neural Information Processing Systems (NIPS)* (2016). 20

[73] LIN, T.-Y., ROYCHOWDHURY, A., AND MAJI, S. Bilinear CNN models for fine-grained visual recognition. In *Proc. IEEE International Conference on Computer Vision (ICCV)* (2015). 21, 95, 97

[74] LONG, J., SHELHAMER, E., AND DARRELL, T. Fully convolutional networks for semantic segmentation. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2015). 34

[75] LONG, M., ZHU, H., WANG, J., AND JORDAN, M. I. Deep transfer learning with joint adaptation networks. *arXiv preprint arXiv:1605.06636* (2016). 61

[76] MA, S., BARGAL, S. A., ZHANG, J., SIGAL, L., AND SCLAROFF, S. Do less and achieve more: Training CNNs for action recognition utilizing action images from the web. *Pattern Recognition* (2017). 66, 73, 108

[77] MAHDISOLTANI, F., BERGER, G., GHARBIEH, W., FLEET, D. J., AND MEMISEVIC, R. Fine-grained video classification and captioning. *CoRR abs/1804.09235* (2018). 127

[78] MAJI, S., RAHTU, E., KANNALA, J., BLASCHKO, M., AND VEDALDI, A. Fine-grained visual classification of aircraft. *arXiv preprint arXiv:1306.5151* (2013). 83, 94

[79] MICONI, T., CLUNE, J., AND STANLEY, K. O. Differentiable plasticity: training plastic neural networks with backpropagation. *arXiv preprint arXiv:1804.02464* (2018). 6, 65

[80] MITTAL, D., BHARDWAJ, S., KHAPRA, M. M., AND RAVINDRAN, B. Recovering from random pruning: On the plasticity of deep convolutional neural networks. *Winter Conference on Applications of Computer Vision* (2018). 6, 65, 66, 73

[81] MNIH, V., HEESS, N., GRAVES, A., ET AL. Recurrent models of visual attention. In *Advances in Neural Information Processing Systems (NIPS)* (2014). 84

[82] MONFORT, M., ZHOU, B., BARGAL, S. A., ANDONIAN, A., YAN, T., RAMAKRISHNAN, K., BROWN, L., FAN, Q., GUTFRUEND, D., VONDRICK, C., ET AL. Moments in time dataset: one million videos for event understanding. *arXiv preprint arXiv:1801.03150* (2018). 117, 118

[83] MORERIO, P., CAVAZZA, J., VOLPI, R., VIDAL, R., AND MURINO, V. Curriculum dropout. In *Proc. IEEE International Conference on Computer Vision (ICCV)* (2017). 6, 19, 20, 65, 70, 72

[84] NG, J. Y.-H., HAUSKNECHT, M., VIJAYANARASIMHAN, S., VINYALS, O., MONGA, R., AND TODERICI, G. Beyond short snippets: Deep networks for video classification. *arXiv preprint arXiv:1503.08909* (2015). 16, 49

[85] ONEATA, D., VERBEEK, J., AND SCHMID, C. The LEAR submission at Thumos 2014. 114

[86] OQUAB, M., BOTTOU, L., LAPTEV, I., AND SIVIC, J. Is object localization for free? - weakly-supervised learning with convolutional neural networks. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2015). 23

[87] OQUAB, M., BOTTOU, L., LAPTEV, I., AND SIVIC, J. Is object localization for free?-weakly-supervised learning with convolutional neural networks. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2015). 18

[88] PAPANDREOU, G., CHEN, L.-C., MURPHY, K., AND YUILLE, A. L. Weakly-and semi-supervised learning of a DCNN for semantic image segmentation. In *Proc. IEEE International Conference on Computer Vision (ICCV)* (2015). 18

[89] PARKHI, O. M., VEDALDI, A., AND ZISSERMAN, A. Deep face recognition. In *Proc. British Machine Vision Conference (BMVC)* (2015). 5

[90] PATHAK, D., KRAHENBUHL, P., AND DARRELL, T. Constrained convolutional neural networks for weakly supervised segmentation. In *Proc. IEEE International Conference on Computer Vision (ICCV)* (2015). 18

[91] PENG, X., USMAN, B., KAUSHIK, N., HOFFMAN, J., WANG, D., AND SAENKO, K. VisDA: The visual domain adaptation challenge. *arXiv preprint arXiv:1710.06924* (2017). 60

[92] PERRONNIN, F., SÁNCHEZ, J., AND MENSINK, T. Improving the fisher kernel for large-scale image classification. In *Proc. European Conference on Computer Vision (ECCV)*. 2010. 15

[93] PETSIUK, V., DAS, A., AND SAENKO, K. RISE: Randomized input sampling for explanation of black-box models. In *Proc. British Machine Vision Conference (BMVC)* (2018). 4, 55, 92

[94] PINHEIRO, P. H., AND COLLOBERT, R. Recurrent convolutional neural networks for scene parsing. In *Proc. International Conference on Learning Representations (ICLR)* (2014). 18

[95] PINHEIRO, P. O., AND COLLOBERT, R. From image-level to pixel-level labeling with convolutional networks. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2015). 18

[96] PLUMMER, B. A., WANG, L., CERVANTES, C. M., CAICEDO, J. C., HOCKENMAIER, J., AND LAZEBNIK, S. Flickr30k entities: Collecting region-to-phrase correspondences for richer image-to-sentence models. In *Proc. IEEE International Conference on Computer Vision (ICCV)* (2015). 114

[97] RAMANISHKA, V., DAS, A., ZHANG, J., AND SAENKO, K. Top-down visual saliency guided by captions. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2017). 8, 23, 99, 114, 115, 116, 117

[98] RAPTIS, M., AND SIGAL, L. Poselet key-framing: A model for human activity recognition. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2013). 15

[99] RENNIE, S. J., GOEL, V., AND THOMAS, S. Annealed dropout training of deep networks. In *Spoken Language Technology Workshop (SLT), 2014 IEEE* (2014), IEEE, pp. 159–164. 19

[100] REYNOLDS, J. H., AND HEEGER, D. J. The normalization model of attention. *Neuron 61*, 2 (2009), 168–185. 17

[101] RIBEIRO, M. T., SINGH, S., AND GUESTRIN, C. Why should i trust you?: Explaining the predictions of any classifier. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining* (2016). 8, 22

[102] RICHARD, A., AND GALL, J. Temporal action detection using a statistical language model. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016). 114

[103] SELVARAJU, R. R., COGSWELL, M., DAS, A., VEDANTAM, R., PARIKH, D., AND BATRA, D. Grad-CAM: Visual explanations from deep networks via gradient-based localization. In *Proc. IEEE International Conference on Computer Vision (ICCV)* (2017). 8, 23, 55, 92, 99

[104] SHOU, Z., CHAN, J., ZAREIAN, A., MIYAZAWA, K., AND CHANG, S.-F. CDC: Convolutional-De-Convolutional networks for precise temporal action localization in untrimmed videos. *arXiv preprint arXiv:1703.01515* (2017). 110

[105] SHOU, Z., WANG, D., AND CHANG, S.-F. Temporal action localization in untrimmed videos via multi-stage CNNs. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016). 114

[106] SIMONYAN, K., VEDALDI, A., AND ZISSERMAN, A. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034* (2013). 4, 17, 18, 22, 99

[107] SIMONYAN, K., AND ZISSERMAN, A. Two-stream convolutional networks for action recognition in videos. In *Advances in Neural Information Processing Systems (NIPS)* (2014). 25, 27, 32, 34, 39, 47, 48, 49

[108] SIMONYAN, K., AND ZISSERMAN, A. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556* (2014). 16, 25, 33, 34, 43, 45, 47, 53, 70, 73, 105

[109] SONG, S., MILLER, K. D., AND ABBOTT, L. F. Competitive hebbian learning through spike-timing-dependent synaptic plasticity. *Nature neuroscience 3*, 9 (2000), 919. 6, 65

[110] SOOMRO, K., ZAMIR, A. R., AND SHAH, M. UCF101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402* (2012). 30, 69, 73, 108

[111] SPRINGENBERG, J. T., DOSOVITSKIY, A., BROX, T., AND RIEDMILLER, M. Striving for simplicity: The all convolutional net. *arXiv preprint arXiv:1412.6806* (2014). 4, 8, 17, 18, 22, 99

[112] SRIVASTAVA, N., HINTON, G., KRIZHEVSKY, A., SUTSKEVER, I., AND SALAKHUTDINOV, R. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research (JMLR) 15*, 1 (2014), 1929–1958. 6, 19, 65, 67

[113] SULTANI, W., AND SHAH, M. What if we do not have multiple videos of the same action? – video action localization using web images. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (June 2016). 17

[114] SUN, C., SHETTY, S., SUKTHANKAR, R., AND NEVATIA, R. Temporal localization of fine-grained actions in videos by domain transfer from web images. *arXiv preprint arXiv:1504.00983* (2015). 16

[115] SUN, M., YUAN, Y., ZHOU, F., AND DING, E. Multi-attention multi-class constraint for fine-grained image recognition. In *Proc. European Conference on Computer Vision (ECCV)* (2018). 21, 83, 84, 94, 95, 96

[116] TRAN, D., BOURDEV, L., FERGUS, R., TORRESANI, L., AND PALURI, M. Learning spatiotemporal features with 3d convolutional networks. In *Proc. IEEE International Conference on Computer Vision (ICCV)* (2015), IEEE, pp. 4489–4497. 16, 49

[117] TREISMAN, A. M., AND GELADE, G. A feature-integration theory of attention. *Cognitive psychology 12*, 1 (1980), 97–136. 17

[118] TSOTSOS, J. K., CULHANE, S. M., WAI, W. Y. K., LAI, Y., DAVIS, N., AND NUFLO, F. Modeling visual attention via selective tuning. *Artificial Intelligence 78*, 1 (1995), 507–545. 4, 17

[119] VENUGOPALAN, S., XU, H., DONAHUE, J., ROHRBACH, M., MOONEY, R., AND SAENKO, K. Translating videos to natural language using deep recurrent neural networks. *North American Chapter of the Association for Computational Linguistics – Human Language Technologies NAACL-HLT* (2015). 105, 115

[120] WAGER, S., WANG, S., AND LIANG, P. S. Dropout training as adaptive regularization. In *Advances in Neural Information Processing Systems (NIPS)* (2013). 19

[121] WAN, L., ZEILER, M., ZHANG, S., LE CUN, Y., AND FERGUS, R. Regularization of neural networks using dropconnect. In *Proc. International Conference on Machine Learning (ICML)* (2013). 19

[122] WANG, D., SHEN, Z., SHAO, J., ZHANG, W., XUE, X., AND ZHANG, Z. Multiple granularity descriptors for fine-grained categorization. In *Proc. IEEE International Conference on Computer Vision (ICCV)* (2015). 21, 95, 97

[123] WANG, H., AND SCHMID, C. Action recognition with improved trajectories. In *Proc. IEEE International Conference on Computer Vision (ICCV)* (2013). 15

[124] WANG, H., AND SCHMID, C. Lear-inria submission for the thumos workshop. In *ICCV Workshop on Action Recognition with a Large Number of Classes* (2013). 15, 47, 48, 49

[125] WANG, H., WU, X., AND JIA, Y. Video annotation via image groups from the web. *IEEE Transactions on Multimedia 16*, 5 (2014), 1282–1291. 16

[126] WANG, L., QIAO, Y., AND TANG, X. Action recognition and detection by combining motion and appearance features. *THUMOS14 Action Recognition Challenge 1*, 2 (2014), 2. 114

[127] WANG, L., QIAO, Y., AND TANG, X. Video action detection with relational dynamic-poselets. In *Proc. European Conference on Computer Vision (ECCV)* (2014). 15

[128] WANG, L., QIAO, Y., AND TANG, X. Action recognition with trajectory-pooled deep-convolutional descriptors. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2015), pp. 4305–4314. 16, 49, 50

[129] WANG, S., AND MANNING, C. Fast dropout training. In *Proc. International Conference on Machine Learning (ICML)* (2013), pp. 118–126. 20

[130] WANG, Y., CHOI, J., MORARIU, V., AND DAVIS, L. S. Mining discriminative triplets of patches for fine-grained classification. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016), pp. 1163–1172. 21, 97

[131] WANG, Y., AND MORI, G. Hidden part models for human action recognition: Probabilistic versus max margin. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI) 33*, 7 (2011), 1310–1323. 15

[132] WEI, Y., FENG, J., LIANG, X., CHENG, M.-M., ZHAO, Y., AND YAN, S. Object region mining with adversarial erasing: A simple classification to semantic segmentation approach. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2017). 7, 85, 88

[133] WEINLAND, D., RONFARD, R., AND BOYER, E. A survey of vision-based methods for action representation, segmentation and recognition. *Computer Vision and Image Understanding (CVIU) 115*, 2 (2011), 224–241. 15

[134] WELINDER, P., BRANSON, S., MITA, T., WAH, C., SCHROFF, F., BELONGIE, S., AND PERONA, P. Caltech-ucsd birds 200. In *Technical Report CNS-TR-2010-001, California Institute of Technology* (2010). 83, 94

[135] WOLFE, J. M. Guided search 2.0 a revised model of visual search. *Psychonomic Bulletin & Review 1*, 2 (1994), 202–238. 17

[136] WU, H., AND GU, X. Towards dropout training for convolutional neural networks. *Neural Networks 71* (2015), 1–10. 19

[137] XU, H., DAS, A., AND SAENKO, K. R-C3D: Region convolutional 3D network for temporal activity detection. In *Proc. IEEE International Conference on Computer Vision (ICCV)* (2017). 110, 114

[138] XU, J., MEI, T., YAO, T., AND RUI, Y. MSR-VTT: A large video description dataset for bridging video and language. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016). 114

[139] XU, K., BA, J., KIROS, R., CHO, K., COURVILLE, A., SALAKHUDINOV, R., ZEMEL, R., AND BENGIO, Y. Show, attend and tell: Neural image caption generation with visual attention. In *Proc. International Conference on Machine Learning (ICML)* (2015). 100

[140] YAO, B., AND FEI-FEI, L. Grouplet: A structured image representation for recognizing human and object interactions. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2010). 30

[141] YAO, B., JIANG, X., KHOSLA, A., LIN, A. L., GUIBAS, L., AND FEI-FEI, L. Human action recognition by learning bases of action attributes and parts. In *Proc. IEEE International Conference on Computer Vision (ICCV)* (2011). 26, 30

[142] YAO, L., TORABI, A., CHO, K., BALLAS, N., PAL, C., LAROCHELLE, H., AND COURVILLE, A. Describing videos by exploiting temporal structure. In *Proc. IEEE International Conference on Computer Vision (ICCV)* (2015). 100

[143] YEUNG, S., RUSSAKOVSKY, O., MORI, G., AND FEI-FEI, L. End-to-end learning of action detection from frame glimpses in videos. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016). 114

[144] YUAN, J., NI, B., YANG, X., AND KASSIM, A. A. Temporal action localization with pyramid of score distribution features. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016). 114

[145] ZEILER, M. D., AND FERGUS, R. Visualizing and understanding convolutional networks. In *Proc. European Conference on Computer Vision (ECCV)* (2014). 4, 8, 17, 18, 22, 99

[146] ZHANG, H., XU, T., ELHOSEINY, M., HUANG, X., ZHANG, S., ELGAMMAL, A., AND METAXAS, D. SPDA-CNN: Unifying semantic part detection and abstraction for fine-grained recognition. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016). 20, 83

[147] ZHANG, J., BARGAL, S. A., LIN, Z., BRANDT, J., SHEN, X., AND SCLAROFF, S. Top-down neural attention by excitation backprop. *International Journal of Computer Vision (IJCV) 126*, 10 (2018), 1084–1102. 7, 55, 86, 92

[148] ZHANG, J., LIN, Z., BRANDT, J., SHEN, X., AND SCLAROFF, S. Top-down neural attention by excitation backprop. In *Proc. European Conference on Computer Vision (ECCV)* (2016). 8, 23, 55, 56, 66, 99, 103, 105, 108, 112

[149] ZHANG, N., DONAHUE, J., GIRSHICK, R., AND DARRELL, T. Part-based R-CNNs for fine-grained category detection. In *Proc. European Conference on Computer Vision (ECCV)* (2014). 20, 83

[150] ZHANG, X., WEI, Y., FENG, J., YANG, Y., AND HUANG, T. Adversarial complementary learning for weakly supervised object localization. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2018). 7

[151] ZHANG, X., XIONG, H., ZHOU, W., LIN, W., AND TIAN, Q. Picking deep filter responses for fine-grained image recognition. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016). 21

[152] ZHAO, B., WU, X., FENG, J., PENG, Q., AND YAN, S. Diversified visual attention networks for fine-grained object classification. *IEEE Transactions on on Multimedia 19*, 6 (2017), 1245–1256. 21, 94, 95, 96

[153] ZHAO, Y., XIONG, Y., WANG, L., WU, Z., TANG, X., AND LIN, D. Temporal action detection with structured segment networks. In *Proc. IEEE International Conference on Computer Vision (ICCV)* (2017). 114

[154] ZHENG, H., FU, J., MEI, T., AND LUO, J. Learning multi-attention convolutional neural network for fine-grained image recognition. In *Proc. IEEE International Conference on Computer Vision (ICCV)* (2017). 21, 83, 84, 94, 95, 97

[155] ZHOU, B., KHOSLA, A., LAPEDRIZA, A., OLIVA, A., AND TORRALBA, A. Object detectors emerge in deep scene CNNs. In *Proc. International Conference on Learning Representations (ICLR)* (2015). 4, 17, 18

[156] ZHOU, B., KHOSLA, A., LAPEDRIZA, A., OLIVA, A., AND TORRALBA, A. Learning deep features for discriminative localization. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016). 4, 8, 18, 22, 55, 99

[157] ZHOU, B., LAPEDRIZA, A., XIAO, J., TORRALBA, A., AND OLIVA, A. Learning deep features for scene recognition using places database. In *Advances in Neural Information Processing Systems (NIPS)* (2014). 4

[158] ZHOU, Y., ZHU, Y., YE, Q., QIU, Q., AND JIAO, J. Weakly supervised instance segmentation using class peak response. *arXiv preprint arXiv:1804.00880* (2018). 7

Curriculum Vitae [redacted]